

**TOUCH SENSOR****Publication number:** JP2001242023**Publication date:** 2001-09-07**Inventor:** AJIOKA YOSHIKI**Applicant:** ECCHANDESU KK**Classification:**

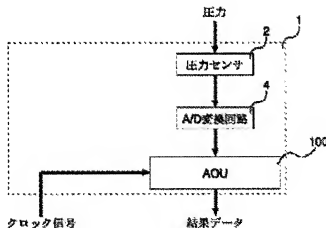
**- international:** G01L5/00; G01B7/00; G01B7/28; G01B21/00;  
G01B21/20; G01L5/00; G01B7/00; G01B7/28;  
G01B21/00; G01B21/20; (IPC1-7): G01L5/00;  
G01B7/00; G01B7/28; G01B21/00; G01B21/20

**- European:****Application number:** JP20000052735 20000229**Priority number(s):** JP20000052735 20000229

Report a data error here

**Abstract of JP2001242023**

**PROBLEM TO BE SOLVED:** To design and manufacture a touch sensor having a pressure sensor, an A/D converting circuit, and an arrangement operation unit arranged into a two-dimensional grid. **SOLUTION:** A function column 1 is provided with the pressure sensor 2, the A/D converting circuit 4, and the arrangement operation unit 100 as shown in Figure 1. The function column 1 is arranged into the two-dimensional grid as shown in Figure 5 and a signal wire is laid between function columns 1 in 4 circumferences. When the respective function columns 1 are provided with oscillation circuits 5 as shown in Figure 2, the signal wire for feeding the same clock signals to all the function columns 1 from the outside can be eliminated. The arrangement operation unit 10 outputs pixels to be outputted, in parallel.



Data supplied from the esp@cenet database - Worldwide



(19)日本特許庁(J P)

## (12) 公開特許公報(A)

(11)特許出願公開番号  
特開2001-242023  
(P2001-242023A)

(43)公開日 平成13年9月7日(2001.9.7)

(51)Int.Cl.	識別記号	F I	テーマコード(参考)
G 0 1 L 5/00	1 0 1	C 0 1 L 5/00	1 0 1 Z 2 F 0 5 1
G 0 1 B 7/00		C 0 1 B 7/00	N 2 F 0 6 3
7/28		7/28	H 2 F 0 6 9
21/00		21/00	D
21/20		21/20	F

審査請求 未請求 請求項の数 5 O L (全 37 頁)

(21)出願番号 特願2000-52735(P2000-52735)

(22)出願日 平成12年2月29日(2000.2.29)

(71)出願人 39805/167

株式会社エッチャンデス  
愛知県蒲郡市中央本町12番7号

(72)発明者 味岡 義明

愛知県蒲郡市中央本町12番7号

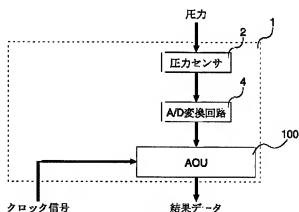
最終頁に続く

(54)【発明の名称】 タッチセンサ

## (57)【要約】

【目的】 圧力センサ、A/D変換回路、配列演算ユニットを二次元格子状に配列したタッチセンサを設計及び製造する。

【構成】 図1に示すように、機能コラム1は圧力センサ2、A/D変換回路4、及び配列演算ユニット100を備えている。また図5に示すように、機能コラム1は二次元格子状に配列され、4近傍の機能コラム1同士の間で信号線が配線される。図2に示すように、各々の機能コラム1が発振回路5を備えた場合、外部から全ての機能コラム1に同一のクロック信号を供給するための信号線が必要なくなる。配列演算ユニット100から、出力すべき画像の画素が並列に出力される。



## 【特許請求の範囲】

【請求項1】 1個の画素に対して画像処理を行う機能コラムが二次元格子状に配列されたタッチセンサであって、前記機能コラムが、

1～16個の圧力センサと、

1～16個のA/D変換回路と、

1～32個の配列演算ユニットとを、備え、前記配列演算ユニットが局所並列画像処理を実行することを特徴とするタッチセンサ。

【請求項2】 請求項1記載の機能コラムに対して、1個の発振回路を備えたことを特徴とするタッチセンサ。

【請求項3】 請求項1及び2記載のタッチセンサに対して、

前記機能コラム内で前記圧力センサが前記A/D変換回路にアナログ信号を送信するように前記信号線が配線されることと、

前記機能コラム内で前記A/D変換回路が前記配列演算ユニットの1個にデジタル信号を送信するように前記信号線が配線されることと、

前記機能コラム内で前記発振回路が前記配列演算ユニットの全てのクロック信号を送信するように前記信号線が配線されることと、

前記配列演算ユニットが2～32個の場合、前記機能コラム内で前記配列演算ユニットの各々が残りの前記配列演算ユニットのうち1～31個と前記デジタル信号を通信するように前記信号線が配線されることと、

前記配列演算ユニットの各々が、二次元格子上で4近傍に位置する前記機能コラムの各々が備える1個の前記配列演算ユニットと前記デジタル信号を通信するように前記信号線が配線されることと、を特徴とするタッチセンサ。

【請求項4】 請求項1、請求項2、及び請求項3記載の機能コラムに対して、圧力センサのうち1～16個の代りに1～16個の温度センサを備えたことを特徴とするタッチセンサ。

【請求項5】 請求項1から請求項4記載の機能コラムに対して、圧力センサのうち1～16個の代りに1～16個及び1～16対の電極を備えたことを特徴とするタッチセンサ。

【発明の詳細な説明】

【0001】

【発明の属する技術の分野】本発明は、画像中の物体の位置及び大きさを検出して物体をパターンマッチングするタッチセンサに関し、詳しくは、圧力センサ、温度センサ及び電極を介して物体の圧力、温度、水分及び電気抵抗を画像化することにより、これらの画像から構成される動画画像中の物体のエッジ情報から、物体の位置及び大きさを検出し、物体のパターンマッチングを行い、物体のマッチング結果、位置及び大きさを出力するものに関する。

【0002】

【従来の技術】近年、圧電素子、圧電振動子、超音波振動子、導電性フィルム、光ファイバ、受光素子、及び電界効果型トランジスタ(MOSFET)等を用いた高特性圧力センサ及び触覚センサが多数開発されるようになってきた(例えば、特開平05-149809、特開平05-215625、特開平05-288619、特開平07-019975、特開平07-115209、特開平09-203671、特開平11-044587、特開平11-088601参照)。また圧力センサ及び触覚センサを二次元格子状に配列して圧力の位置及び大きさを検出するタッチセンサも開発されている(例えば、特開平05-149809、特開平05-215625、特開平06-011403、特開平06-018216、特開平07-019975、特開平07-190870、特開平11-118460、特表平11-506559参照)。その他に、焦電素子、赤外線受光素子、光導波路、光シャッター素子、サーミスタ及び熱電対等の温度センサを用いたもの(例えば、特開平05-03346、特開平07-181087、特開平08-076174、特開平08-122246、特開平08-320258、特開平09-033215、特開平09-035038、特開平10-181098、特開平11-354829参照)、及び電極を用いたもの(例えば、特開平05-010826、特開平10-240435、特開平10-314148、特開平2000-005147参照)も開発されている。しかしながら上記のようなタッチセンサでは、各種センサと、入力刺激の位置、大きさ及び動きを検出し入力刺激に対してパターンマッチングをする装置とが独立に存在するため、各種センサと装置の間の通信がボトルネックになり、入力刺激が二次元であることを装置が利用できなくなったり、さらには配線及び付加回路のために装置自体が大きくなってしまったといった問題が生じる。

【0003】一方で、本発明者は既に、エッジ情報生成装置(特願平11-145638参照)、エッジ情報形成装置(特願平11-209738参照)、物体/背景分離装置(特願平10-257327参照)、位置/大きさ検出装置(特願平11-250990参照)、領域正規化装置(特願平11-250986参照)、及び視覚装置(特願平11-253634参照)を開発している。これらの装置は統一的なアーキテクチャ上で画素単位で並列に画像処理を行い、ノイズの多い実環境で移動物体及び静止物体のエッジ情報を生成し、移動物体及び静止物体の領域を背景から分離し、移動物体及び静止物体の領域の位置及び大きさを検出し、移動物体及び静止物体の領域を画像サイズに正規化してパターンマッチングをすることができる。さらに視覚装置を電荷結合素子(CCD)及びフォトダイオードと共に大規模集積回路(LSI)に組み込むことにより、高機能イメージセ

ンサーができる(特願平11-369220参照)。

【0004】これらのことを考慮すると、LSI実装技術を用いて、圧力センサ、温度センサ及び電極と共に視覚装置の局所並列画像処理をLSIに実装することができれば、実時間で移動物体及び静止物体のマッチング結果、位置及び大きさを出力することができる高機能なタッチセンサを製造できると期待される。

【0005】

【発明が解決しようとする課題】そこで、請求項記載の本発明は、二次元格子状に配列した圧力センサ、温度センサ及び電極等の各種センサ、A/D変換回路、及び配列演算ユニットをLSIに実装し、局所並列画像処理を実行することにより、移動物体及び静止物体を射影してエッジ情報を生成し、移動物体及び静止物体の領域を背景から分離し、移動物体及び静止物体の領域の位置及び大きさを検出し、移動物体及び静止物体の領域を画像サイズに正規化してパターンマッチングをすることを目的とする。

【0006】

【課題を解決するための手段】請求項1の発明は、1個の画素に対して画像処理を行う機能コラムが二次元格子状に配列されたタッチセンサであって、前記機能コラムが、1～16個の圧力センサと、1～16個のA/D変換回路と、1～32個の配列演算ユニットと、を備え、前記配列演算ユニットが局所並列画像処理を実行することと特徴とするタッチセンサである。本発明は、圧電素子、圧電振動子、超音波振動子、導電性フィルム、光ファイバ、受光素子、及び電界効果型トランジスタ(MOSFET)等から構成される前記圧力センサを用いて圧力を検知することにより、前記圧力センサからアナログ信号を取り出し、前記A/D変換回路を用いて前記アナログ信号をデジタル信号に変換し、前記デジタル信号を前記配列演算ユニットに入力する。前記配列演算ユニットは、画像記憶手段、画像振動手段、エッジ情報生成手段、エッジ情報形成手段、物体/背景分離手段、位置/大きさ検出手段、領域正規化手段、及びパターンマッチング手段を用いて前記局所並列画像処理を実行し、前記デジタル信号によって構成されるデジタル画像中に投影された移動物体及び静止物体の位置及び大きさを検出し、前記移動物体及び前記静止物体のパターンマッチングを行う。前記配列演算ユニットは、前記画像記憶手段、前記画像振動手段、前記エッジ情報生成手段、前記エッジ情報形成手段、前記物体/背景分離手段、前記位置/大きさ検出手段、前記領域正規化手段、及び前記パターンマッチング手段の処理結果の各々を、前記デジタル画像、振動画像、粗エッジ情報画像、形成エッジ情報画像、物体領域画像、重複情報画像、正規化画像、及びマッチング結果画像の画素として、画像保持手段を用いて出力することができる。したがって前記タッチセンサの設計及び製造に関する諸問題が好適に解決される。

【0007】請求項2の発明は、請求項1記載の機能コラムに対して、1個の発振回路を備えたことを特徴とするタッチセンサである。本発明は、前記機能コラムの全てが独立した前記発振回路を備えることにより、大規模集積回路に対してクロックスキューなどの遅延時間問題及び配線問題を回避することができる。したがって前記タッチセンサの設計及び製造に関する諸問題が好適に解決される。

【0008】請求項3の発明は、請求項1及び2記載のタッチセンサに対して、前記機能コラム内で前記圧力センサが前記A/D変換回路にアナログ信号を送信するように前記信号線が配線されることと、前記機能コラム内で前記A/D変換回路が前記配列演算ユニットの1個にデジタル信号を送信するように前記信号線が配線されることと、前記機能コラム内で前記発振回路が前記配列演算ユニットの全てにクロック信号を送信するように前記信号線が配線されることと、前記配列演算ユニットが2～32個の場合、前記機能コラム内で前記配列演算ユニットの各々が残りの前記配列演算ユニットのうち1～31個と前記デジタル信号を送信するように前記信号線が配線されることと、前記配列演算ユニットの各々が、二次元格子状上4近傍に位置する前記機能コラムの各々が備える1個の前記配列演算ユニットと前記デジタル信号を送信するように前記信号線が配線されることと、と特徴とするタッチセンサである。本発明は、前記信号線が前記機能コラム内と、前記二次元格子上で互いに前記4近傍に位置する前記機能コラム間で配線される。つまり前記機能コラムが互いに前記4近傍に位置しない前記機能コラム間で前記信号線が配線されないで、大規模集積回路の配線パターンは単純化され、しかも其距離配線による遅延時間を考慮する必要がなくなる。したがって前記タッチセンサの設計及び製造に関する諸問題が好適に解決される。

【0009】請求項4の発明は、請求項1、請求項2、及び請求項3記載の機能コラムに対して、圧力センサのうち1～16個の代りに1～16個の温度センサを備えたことを特徴とするタッチセンサである。本発明は、焦電素子、赤外線受光素子、光導波路、光シャット素子、サーミスタ及び熱電対等から構成される前記温度センサを用いて温度を検知することにより、前記温度センサからアナログ信号を取り出し、前記A/D変換回路を用いて前記アナログ信号をデジタル信号に変換し、前記デジタル信号を前記配列演算ユニットに入力する。圧力センサ及び温度センサが1つの機能コラムに含まれても良い。これにより前記タッチセンサの性能は向上する。前記配列演算ユニットは、画像記憶手段、画像振動手段、エッジ情報生成手段、エッジ情報形成手段、物体/背景分離手段、位置/大きさ検出手段、領域正規化手段、及びパターンマッチング手段を用いて前記局所並列画像処理を実行し、前記デジタル信号によって構成されるデジ

たる画像中に射影された移動物体及び静止物体の位置及び大きさを検出し、前記移動物体及び前記静止物体のパターンマッチングを行う。前記配列演算ユニットは、前記画像記憶手段、前記画像振動手段、前記エッジ情報生成手段、前記エッジ情報形成手段、前記物体／背景分離手段、前記位置／大きさを検出手段、前記領域正規化手段、及び前記パターンマッチング手段の処理結果の各々を、前記デジタル画像、振動画像、粗エッジ情報画像、形成エッジ情報画像、物体領域画像、重複情報画像、正規化画像、及びマッチング結果画像の画素として、画像保持手段を用いて出力することができる。したがって前記タッチセンサの設計及び製造に関する諸問題が好適に解決される。

【0010】請求項5の発明は、請求項1から請求項4記載の機能コラムに対して、圧力センサのうち1〜16個の代りに1〜16個及び1〜16対の電極を備えたことを特徴とするタッチセンサである。本発明は、前記電極を用いて電気及び抵抗を検知することにより、前記電極からアナログ信号を取り出し、前記A/D変換回路を用いて前記アナログ信号をデジタル信号に変換し、前記デジタル信号を前記配列演算ユニットに入力する。圧力センサ、温度センサ及び電極が1つの機能コラムに含まれても良い。これにより前記タッチセンサの性能は向上する。前記配列演算ユニットは、画像記憶手段、画像振動手段、エッジ情報生成手段、エッジ情報形成手段、物体／背景分離手段、位置／大きさを検出手段、領域正規化手段、及びパターンマッチング手段を用いて前記局所並列画像処理を実行し、前記デジタル信号によって構成されるデジタル画像中に射影された移動物体及び静止物体の位置及び大きさを検出し、前記移動物体及び前記静止物体のパターンマッチングを行う。前記配列演算ユニットは、前記画像記憶手段、前記画像振動手段、前記エッジ情報生成手段、前記エッジ情報形成手段、前記物体／背景分離手段、前記位置／大きさを検出手段、前記領域正規化手段、及び前記パターンマッチング手段の処理結果の各々を、前記デジタル画像、振動画像、粗エッジ情報画像、形成エッジ情報画像、物体領域画像、重複情報画像、正規化画像、及びマッチング結果画像の画素として、画像保持手段を用いて出力することができる。したがって前記タッチセンサの設計及び製造に関する諸問題が好適に解決される。

・【0011】

【発明の実施の形態】以下、本発明のタッチセンサの実施形態を挙げ、図面を参照して説明する。

【0012】まず請求項1及び2記載の発明は各々図1及び2に示すような機能コラム1を二次元格子状に配列することによって実現される。なお図1及び2において、配列演算ユニット100をAOUと略記する。圧力センサ2及びA/D変換回路4の対の数が2個以上の場合、この対は図1及び2の圧力センサ2及びA/D変換

回路4の対に対して並列に配置される。つまり圧力センサ2及びA/D変換回路4の対は実装面に対して二次元に配列される。配列演算ユニット100の数が2個以上の場合、図1及び2の配列演算ユニット100を根とすると本構造のように配置される。具体的には、機能コラム1は図3及び4に示すようになる。図3に示した機能コラム1はタッチセンサに触れた物体の位置及び接触面積を検知するためのものであり、4に示した機能コラム1はタッチセンサに触れた移動物体の位置及び接触面積を検知するためのものである。なお、図1及び2に示された1個の配列演算ユニット100でも、プログラムによって複数の手段を実現できる。

【0013】請求項3記載の発明では、図1から4までに示すような機能コラム1を二次元格子状に配列した場合、図5に示すように信号線が配線される。なお図5において、機能コラム1をFCと略記する。また電源線及びクロック信号は省略されている。図5に示された信号線は実際には配列演算ユニット100に接続されている。図5から明らかなように、機能コラム1間の信号線は、他の機能コラム1間の信号線と交差しない。しかも配列演算ユニット100は同じ画像処理手段を実現する配列演算ユニット100とだけ通信をするので、たとえ機能コラム1に複数の配列演算ユニット100が含まれるとしても、これらの信号線は機能コラム1間では交差しない。図6に示すように、機能コラム1中の全ての配列演算ユニット100が三次元大規模集積回路（三次元VLSI）によって垂直に並ぶように実装されれば、機能コラム1間の信号線は配列演算ユニット100間ですら交差しない。ただし図6では、同じ高さ位置する配列演算ユニット100は同じ画像処理手段を実現するものとする。したがってイメージセンサの設計及び実装は極めて容易になる。

【0014】請求項4及び5記載の発明は、請求項1、2、及び3記載の機能コラム1において、圧力センサ2の代りに温度センサ及び電極を実装したものである。したがって配列演算ユニット100及び配線に関して特に変更はない。ただし相補性金属酸化膜半導体（CMOS）によって実現される温度センサ及び電極を用いる場合、設計工程及び製造工程が簡単になる。

【0015】ここまでは、請求項1から5記載のタッチセンサに対して、二次元格子状に配列された機能コラム1の構成と、配列演算ユニット100間の信号線の配線について説明してきた。配列演算ユニット100は、これらの信号線を介して4近傍の配列演算ユニット100と画像データを通信することにより、以下のような画像処理手段を実行することができる。

【0016】配列演算ユニット100が実行する画像記憶手段12、画像振動手段、エッジ情報生成手段14、エッジ情報形成手段15、物体／背景分離手段16、領域正規化手段27、位置／大きさを検出手段17、パター

ンマッチング手段29、及び画像保持手段38、は二次元格子状に配列された配列演算ユニット100 (ARRAY OPERATION UNIT) から構成されるデータ処理装置110を用いることにより実施することができる。そこで以下では、配列演算ユニット100を利用したデータ処理装置110の実施形態を挙げ、図面を参照して説明する。

【0017】まず配列演算ユニット100は、入力画像の1つの画素とその近傍画素を用いることにより、出力画像の1つの画素を生成する。そこで図7に示したように、配列演算ユニット100を入力画像のサイズに合わせて二次元格子状に配列したデータ処理装置110を用いることにより、データ処理装置110は入力画像から出力画像を生成することができる。なお図7において、配列演算ユニット100をAOUと略記する。したがって配列演算ユニット100のアルゴリズムを示すことにより、データ処理装置110の画像処理を示すことがで

$$x = \{x_{ijk} | x_{ijk} \text{ is value at } p(i, j, k), 1 \leq i \leq w, 1 \leq j \leq h, 1 \leq k \leq b\}$$

【0020】

【数2】

$$y = \{y_{ijk} | y_{ijk} \text{ is value at } p(i, j, k), 1 \leq i \leq w, 1 \leq j \leq h, 1 \leq k \leq b\}$$

【0021】

【数3】

$$w = \{w_{ijk} | w_{ijk} \text{ is value at } p(i, j, k), 1 \leq i \leq w, 1 \leq j \leq h, 1 \leq k \leq b\}$$

【0022】まず前記画像の各帯域画素値に対する点処理に関する関数について以下で説明する。

【0023】画像 $x$ を二値画像に変換する場合、数式4に従って帯域画素値を二値化する。

【0024】

【数4】

$$\Phi_{ijk}(x) = \begin{cases} 1 & \text{if } x_{ijk} > 0, \\ 0 & \text{otherwise.} \end{cases}$$

【0025】画像 $x$ を帯域最大値画像に変換する場合、数式5に従って1行1列の画素の各帯域の値のうち最大値を選択する。なお前記帯域最大値画像は単帯域画像となるので、便宜上帯域数1の前記画像として取り扱うこととする。したがって関数 $B_{ij1}(x)$ の第3添字は1となっている。

【0026】

【数5】

$$B_{ij1}(x) = \max_k \{x_{ijk}\}$$

【0027】画像 $x$ が二値画像であるとして、画像 $x$ を反転させる場合、数式6に従って計算する。

【0028】

【数6】

$$I_{ijk}(x) = 1 - x_{ijk}$$

【0029】画像 $x$ の位置 $p(i, j, k)$ における対

きる。そこで配列演算ユニット100のアルゴリズムを示すために、画像記憶手段12、画像振動手段、エッジ情報生成手段14、エッジ情報形成手段15、物体/背景分離手段16、領域正規化手段27、位置/大きさ検出手段17、パターンマッチング手段29、及び画像保持手段38、で用いる数式について説明する。

【0018】幅 $w$ 、高さ $h$ 、帯域数 $b$ の任意の $2^n$ 階調画像を $x, y, w$ とすると、 $x, y, w$ は各々位置 $p(i, j, k)$ の帯域画素値 $x_{ijk}, y_{ijk}, w_{ijk}$ を用いて数式1、2及び3のように表される。なおアンダーラインが付された文字はベクトルを示す。また $n$ は非負の整数、 $w, h, b, i, j, k$ は自然数である。

【0019】

【数1】

数変換は数式7に従って行われる。ここで $e$ はオフセットであり、自然対数関数が出力する値が有効範囲に入るようにするために使われるので、一般に $e = 1$ で十分である。この対数化により帯域画素値同士の除算を減算にすることができる。また画像 $x$ が $2^n$ 階調のデジタル画像111であるとする、帯域数に関わらず $2^n$ 個の要素を含むルックアップテーブルをメモリ102上を持つならば、毎回自然対数関数を計算する必要もなく、標準的な対数表を持つ必要もなくなる。

【0030】

【数7】

$$I_{ijk}(x) = \ln(x_{lmk} + e)$$

【0031】さて、画像の位置 $p(i, j, k)$ における $q$ 近傍の位置の集合 $P_{ijk}(q)$ は数式8によって表される。ただし $q$ は4、8、24、48、80、120、 $(2r+1)^2 - 1$ と続く数列であり、 $r$ は自然数である。なお画像サイズをはみ出した位置が集合 $P_{ijk}(q)$ に含まれる場合には、特に指定がない限り位置 $p(i, j, k)$ を代用するものとする。またこれ以外のときは、指定に従い、画素値が0に相当し、しかも画像に含まれない架空の位置を代用する。これにより辺縁処理は自動的に行われる。したがって集合 $P_{ijk}(q)$ の要素の数 $N_{ijk}$ は常に $q$ となる。

【0032】

【数8】

$$P_{ijk}(q) = \begin{cases} \{p(i+1, j, k), p(i, j+1, k), p(i-1, j, k), p(i, j-1, k)\} & \text{if } q = 4, \\ \{p(l, m, k) \mid i-r \leq l \leq i+r, j-r \leq m \leq j+r, p(l, m, k) \neq p(i, j, k)\} & \text{if } q = (2r+1)^2 - 1. \end{cases}$$

【0033】そこで次に画像の各帯域画素値に対する最大8近傍の近傍処理に関する関数及びオペレータについて以下で説明する。

【0034】画像xの位置p(i, j, k)における振動は数式9に従って行われる。ここで位置p(i, j, k)のq近傍の中から1つの位置だけを選択する方法によって画像単位で振動させるか画素単位で振動させるかを決定することができる。もし画像xの全ての位置にお

$$x_{ijk}(x) = x_{lmk} \quad \text{for only one of } p(l, m, k) \in P_{ijk}(q)$$

【0036】画像xの位置p(i, j, k)における平滑化は数式10に従って行われる。ただしint(v)は実数vの小数点以下切り捨てを意味するものとする。もし画像xの帯域画素値が整数値であるならば、ハードウェアの実装時にN<sub>ijk</sub>=4のときx<sub>lmk</sub>の総和に対して右シフト命令を2回、N<sub>ijk</sub>=8のときx<sub>lmk</sub>の総和に対して右シフト命令を3回実行するような回路に変更することにより、除算を実行する回路を省くことができる。

【0037】

【数10】

$$S_{ijk}(x) = \text{int} \left( \frac{1}{N_{ijk}} \sum_{p(l, m, k) \in P_{ijk}(q)} x_{lmk} \right)$$

【0038】ラプラシアンは計算であるが、これは数式11に示すように単なる2階差分オペレータである。8近傍の方がノイズの微妙な変化を捉えてゼロ点およびゼロ交差が多くなり、本発明には向いている。ただしN<sub>ijk</sub>が4か8であるので、もしハードウェアの実装時にN<sub>ijk</sub>=4のときx<sub>lmk</sub>に対して左シフト命令を

$$Z_{ijk}(x) = \begin{cases} 1 & \text{if } x_{ijk} \leq 0 \text{ and } x_{lmk} \geq 0 \text{ for } p(l, m, k) \in P_{ijk}(q), \\ 0 & \text{otherwise.} \end{cases}$$

【0042】画像xが任意の二値画像であるとして、画像xのうち孔が空いている画素を埋める場合には、数式13に従い計算する。ここでfは埋めるべき孔の大きさを表すパラメータであり、一般にはf=1で十分であ

$$F_{ijk}(x) = \begin{cases} 1 & \text{if } \sum_{p(l, m, k) \in P_{ijk}(q)} x_{lmk} + f \geq N_{ijk}, \\ x_{ijk} & \text{otherwise.} \end{cases}$$

【0044】画像xが任意の二値画像であるとして、画像xのうち孤立点ないし孤立孔を削除する場合には、数式14に従い計算する。なお4近傍の場合にはその性質上対角線を検知することができないので、極力8近傍に

いて、全く同じ方法によりq近傍の中から1つを選択すれば、画像xは画像単位で振動する。一方で画像xの各々の位置において、疑似乱数などを用いてランダムにq近傍の中から1つを選択すれば、画像xは画素単位で振動する。

【0035】

【数9】

2回、N<sub>ijk</sub>=8のときx<sub>lmk</sub>に対して左シフト命令を3回実行するような回路に変更することにより、実数の乗算を実行する回路を省くことができる。

【0039】

【数11】

$$V_{ijk}^2(x) = \sum_{p(l, m, k) \in P_{ijk}(q)} x_{lmk} - N_{ijk} x_{ijk}$$

【0040】ラプラシアンによって求められた値からゼロ点を見付ける方法として、従来は正から負に変化する画素を見付けていたが、本発明では数式12に従い、負から正にゼロ交差する画素の他に、負からゼロやゼロから正などゼロ点が経由したり、ゼロが継続する画素を見付けるようにする。本発明では、数式12が見付けたゼロ点はエッジがある場所ではなく、ノイズがある場所、つまりエッジがない場所になる。また数式12により実数値の二値化も同時に行っている。

【0041】

【数12】

る。なお4近傍の場合にはその性質上対角線を検知することができないので、極力8近傍にした方がよい。

【0043】

【数13】

した方がよい。

【0045】

【数14】



$$A_{ijk}(x) = \begin{cases} 0 & \text{if } x_{ijk} = 1 \text{ and } \sum_{p(l,m,k) \in P_{ijk}(q)} x_{lmk} = 0, \\ 1 & \text{if } x_{ijk} = 0 \text{ and } \sum_{p(l,m,k) \in P_{ijk}(q)} x_{lmk} = N_{ijk}, \\ x_{ijk} & \text{otherwise.} \end{cases}$$

【0046】画像  $x$  が任意の二値画像であるとして、画像  $x$  のうち線幅が 1 である画素を検知するために、4 近傍画素を用いて数式 15 に従い計算する。

【0047】  
【数 15】

$$f_{ijk}(x) = \begin{cases} x_{ijk} & \text{if } x_{i-1jk} + x_{i+1jk} = 0 \text{ or } x_{ij-1k} + x_{ij+1k} = 0, \\ 0 & \text{otherwise.} \end{cases}$$

【0048】2つの画像  $x$ 、 $y$  が任意の二値画像であり、画像  $y$  が画像  $x$  のうち線幅が 1 である画素を検知した画像であるとする、画像  $x$  のうち線幅が 1 である画素の線幅を拡張するために、4 近傍画素を用いて数式 1

6 に従い計算する。  
【0049】  
【数 16】

$$K_{ijk}(x, y) = \begin{cases} 1 & \text{if } y_{i-1jk} + y_{i+1jk} + y_{ij-1k} + y_{ij+1k} > 0, \\ x_{ijk} & \text{otherwise.} \end{cases}$$

【0050】そこで数式 15 の線幅検知と数式 16 の線幅拡張を用いると、数式 17 に従い二値画像の線幅補完を簡単に記述することができる。

【0051】  
【数 17】

$$C_{ijk}(x) = K_{ijk}(x, I(x))$$

【0052】次に画像の各帯域画素値に対する近傍処理に関する関数及びオペレータについて以下で説明する。

【0053】2つの画像  $x$ 、 $y$  がある場合、これらの画像の最大値画像は数式 18 に従って計算される。

【0054】  
【数 18】

$$M_{ijk}(x, y) = \begin{cases} x_{ijk} & \text{if } x_{ijk} \geq y_{ijk}, \\ y_{ijk} & \text{otherwise.} \end{cases}$$

【0055】2つの画像  $x$ 、 $y$  がある場合、これらの画像の差分は数式 19 に従って計算される。

【0056】  
【数 19】

$$D_{ijk}(x, y) = x_{ijk} - y_{ijk}$$

【0057】ここで数式 11 のラプラシアンと数式 19 の差分を用いると、数式 20 に従い画像の鮮鋭化を簡単に記述することができる。

【0058】  
【数 20】

$$E_{ijk}(x) = D_{ijk}(x, \nabla_{ijk}^2 x)$$

【0059】2つの画像  $x$ 、 $y$  があり、画像  $y$  が単帯域二値画像である場合、数式 21 に従い、画像  $y$  の帯域画素値を用いて画像  $x$  の各帯域画素値をマスクすることができる。

【0060】  
【数 21】

$$O_{ijk}(x, y) = x_{ijk} y_{ij1}$$

【0061】2つの画像  $x$ 、 $y$  があり、画像  $x$  と  $y$  が二値画像である場合、数式 22 に従い、画像  $x$  を基に画像  $y$  を整形することができる。

【0062】  
【数 22】

$$Q_{ijk}(x, y) = \begin{cases} x_{ijk} & \text{if } y_{ijk} + \sum_{p(l,m,k) \in P_{ijk}(q)} y_{lmk} > 0, \\ 0 & \text{otherwise.} \end{cases}$$

【0063】2つの画像  $x$ 、 $y$  があり、画像  $y$  が二値画像である場合、数式 23 に従い、画像  $y$  で指定されなかった画像  $x$  の帯域画素値を、画像  $x$  の帯域画素値の近傍のうち画像  $y$  で指定された画像  $x$  の帯域画素値の平均値

$$V_{ijk}(x, y) = \begin{cases} \text{int} \left( \frac{\sum_{p(l,m,1) \in P_{ij1}(q)} x_{lm1} y_{lm1}}{\sum_{p(l,m,1) \in P_{ij1}(q)} y_{lm1}} \right) & \text{if } y_{ij1} = 0 \text{ and } \sum_{p(l,m,1) \in P_{ij1}(q)} y_{lm1} > 0, \\ x_{ijk} & \text{otherwise.} \end{cases}$$

で補間する。ただし  $\text{int}(v)$  は実数  $v$  の小数点以下切り捨てを意味するものとする。

【0064】  
【数 23】

【0065】さて本発明では、画素の位置や移動量など

も画像データのように扱うことで処理を単純にしてい

る。これを位置の画像化と呼ぶ。以下では画像化に関する幾つかの関数及びオペレータについて説明する。

【0066】まず位置  $p(1, m, o)$  の  $1, m, o$  各々の値を画像データとして帯域画素値に変換するオペレータを  $\#$  とし、変換された帯域画素値を  $\#p(1, m, o)$  とする。次に帯域画素値が位置  $p(i, j, k)$  から位置  $p(i+1, j+m, k+o)$  へ移動する場合を考える。このとき帯域画素値の移動量は位置  $p(1, m, o)$  として表されるものとする。つまり移動量をある位置からのベクトルと見なすことができる。最後に帯域画素値から位置を取り出すオペレータを  $\#^{-1}$  とする。したがって  $\#^{-1} \#p(1, m, o) = p(1, m, o)$  となる。

【0067】そこで数式24に従い、移動量  $p(i,$

$$G_{ij1}(x) = p\left(\sum_{p(i,m,o) \in P_{ij1}(g)} (i-j)x_{im1}, \sum_{p(i,m,o) \in P_{ij1}(g)} (m-j)x_{im1}, 0\right)$$

【0071】移動量  $p(i, j, k)$  から、数式26及び27に従い8近傍内への移動量を計算し、移動量画像に画像化することができる。なお数式27は、画像の離散化により数式26では対応しきれない場合にのみ利用

$$\Theta(p(i, j, k)) = \begin{cases} \#p(1, 0, k) & \text{if } i > 0, |j| < |i|/2, \\ \#p(1, -1, k) & \text{if } i > 0, j < 0, |i|/2 \leq |j| \leq 2|i|, \\ \#p(0, -1, k) & \text{if } j < 0, 2|i| < |j|, \\ \#p(-1, -1, k) & \text{if } i < 0, j < 0, |i|/2 \leq |j| \leq 2|i|, \\ \#p(-1, 0, k) & \text{if } i < 0, |j| < |i|/2, \\ \#p(-1, 1, k) & \text{if } i < 0, j > 0, |i|/2 \leq |j| \leq 2|i|, \\ \#p(0, 1, k) & \text{if } j > 0, 2|i| < |j|, \\ \#p(1, 1, k) & \text{if } i > 0, j > 0, |i|/2 \leq |j| \leq 2|i|, \\ \#p(0, 0, k) & \text{otherwise.} \end{cases}$$

【0073】

【数27】

$$\Theta'(p(i, j, k)) = \begin{cases} \#p(1, 0, k) & \text{if } i > 0, |j| < |i|/2, \\ \#p(1, 0, k) & \text{if } i > 0, j < 0, |i|/2 \leq |j| \leq 2|i|, \\ \#p(0, 1, k) & \text{if } i < 0, j > 0, |i|/2 \leq |j| \leq 2|i|, \\ \#p(0, 1, k) & \text{if } j > 0, 2|i| < |j|, \\ \#p(1, 1, k) & \text{if } i > 0, j > 0, |i|/2 \leq |j| \leq 2|i|, \\ \#p(0, 0, k) & \text{otherwise.} \end{cases}$$

【0074】したがって数式25、26及び27を用いると、数式28及び29に従い、単帯域二値画像  $x$  の重心方向への移動量画像の帯域画素値を簡単に記述することができる。なお移動量画像の帯域数は1となる。

【0075】

【数28】

$$\Delta_{ij1}(x) = \Theta(G_{ij1}(x))$$

【0076】

【数29】

$$\Delta'_{ij1}(x) = \Theta'(G_{ij1}(x))$$

$j, k$ ) を幅方向と高さ方向で表される平面内で180度反対方向に向けることができる。

【0068】

【数24】

$$T(p(i, j, k)) = p(-i, -j, k)$$

【0069】画像  $x$  があり、画像  $x$  が単帯域二値画像である場合、画像  $x$  の位置  $p(i, j, 1)$  における重心位置への移動量は数式25に従い計算される。なお、本来重心を計算する際には除算を行う必要があるが、8近傍内への移動量を計算する際に除算は相殺されてしまうので、数式25では除算が省かれている。

【0070】

【数25】

する。

【0072】

【数26】

【0077】一方で数式24を用いると重心位置の反対位置を求めることができるので、数式30に従い、単帯域二値画像  $x$  の重心と反対方向への移動量画像の帯域画素値を簡単に記述することができる。なお移動量画像の帯域数は1となる。

【0078】

【数30】

$$R_{ij1}(x) = \Theta(T(G_{ij1}(x)))$$

【0079】2つの画像  $x, y$  があり、画像  $y$  が移動量画像である場合、数式31に従い、画像  $y$  で指し示された移動位置に画像  $x$  の帯域画素値を移動した後、同じ帯

域画素に移動した帯域画素値の合計を濃淡画像にすることができ、

$$\Gamma_{ijk}(x, y) = \sum x_{imk} \text{ for } p(l, m, 1) \in P_{ij1}(q) \text{ and } \#^{-1}y_{lm1} = p(i-l, j-m, 0).$$

【0081】そこで数式4、28、29及び31を用いることにより、数式32又は数式33に従い、単帯域濃淡画像xを近傍の重心方向に移動した後、同じ帯域画素に移動した帯域画素値の合計を簡単に記述することができる。

【0082】

【数32】

$$\Lambda_{ij1}(x) = \Gamma_{ij1}(x, \Delta(\Phi(x)))$$

【0083】

【数33】

$$\Lambda'_{ij1}(x) = \Gamma_{ij1}(x, \Delta'(\Phi(x)))$$

$$H_{ij1}(x, y) = \begin{cases} 1 & \text{if } x_{ij1} = 0 \text{ and } \#^{-1}y_{lm1} = p(i-l, j-m, 0) \\ & \text{for only one of } p(l, m, 1) \in P_{ij1}(q), \\ 0 & \text{otherwise.} \end{cases}$$

【0086】3つの画像x、y、wがあり、画像yが移動可能画像であり、画像wが移動量画像である場合、数式35に従い画像xの帯域画素値を移動することができ

$$T_{ijk}(x, y, w) = \begin{cases} x_{imk} & \text{if } y_{ij1} = 1 \text{ and } \#^{-1}w_{lm1} = p(i-l, j-m, 0) \\ & \text{for } \exists p(l, m, 1) \in P_{ij1}(q), \\ 0 & \text{if } y_{lm1} = 1 \text{ and } \#^{-1}w_{ij1} = p(l-i, m-j, 0) \\ & \text{for } \exists p(l, m, 1) \in P_{ij1}(q), \\ x_{ijk} & \text{otherwise.} \end{cases}$$

【0088】そこで数式30、数式34及び数式35を用いると、数式36に従い、二値画像yから計算される重心位置と反対方向に画像xの帯域画素を移動することで得られる画像の帯域画素値を簡単に記述することができる。

【0089】

【数36】

$$U_{ijk}(x, y) = T_{ijk}(x, H(y, R(y)), R(y))$$

【0090】そこで数式1から数式36までを用いることにより、画像記憶手段12、画像振動手段、エッジ情報生成手段14、エッジ情報形成手段15、領域正規化手段27、位置/大きさ検出手段17、及び画像保持手段38、をインストールするデータ処理装置110の全ての配列演算ユニット100のアルゴリズムを記述することができる。以下では、データ処理装置110中の任意の配列演算ユニット100のアルゴリズムを用いて、画像記憶手段12、画像振動手段、エッジ情報生成手段14、エッジ情報形成手段15、領域正規化手段27位置/大きさ検出手段17、及び画像保持手段38、を説明する。

【0091】データ処理装置110によって実現される

【0080】

【数31】

【0084】2つの画像x、yがあり、画像xが二値画像で、画像yが移動量画像である場合、画像xの各帯域画素値の移動先の位置を求めることができるので、移動先が重複する帯域画素値を見つけることができる。そこで画像xの各帯域画素値の移動先が重複することなく、しかも移動する各帯域画素値が存在することを表す移動可能画像の帯域画素値は、数式34に従い生成される。なお移動可能画像の帯域数は1となる。

【0085】

【数34】

る。

【0087】

【数35】

画像記憶手段12がデジタル画像111を記憶するために、二次元格子状に配列された配列演算ユニット100は同期して並列に動作する。格子上i行j列に配置された配列演算ユニット100をAOU<sub>i,j</sub>とすると、AOU<sub>i,j</sub>のアルゴリズムは図8のようになる。

【0092】ステップ1201で、AOU<sub>i,j</sub>を格子上のi行j列に配置する。これは論理的であれ物理的であれ、AOU<sub>i,j</sub>の近傍を決定するために必要である。

【0093】ステップ1202で、AOU<sub>i,j</sub>の近傍や変数の初期値を設定する。

【0094】ステップ1203で、順次入力されるデジタル画像111が無くなったかどうか判断する。もしデジタル画像111が無ければ(ステップ1203: YES)、アルゴリズムを終了する。もしデジタル画像111があれば(ステップ1203: NO)、ステップ1204に移行する。ただし特定の画像サイズのみに対して配列演算ユニット100をインストールする場合には、無限ループにしても良い。

【0095】ステップ1204で、デジタル画像111が準備されるまで入力待ちをする。

【0096】ステップ1205で、デジタル画像111

の  $i$  行  $j$  列の画素を帯域数分入力する。このため  $AOU_{ij}$  は少なくとも帯域数分の画像データを記憶するメモリ 102 を必要とする。

【0097】ステップ 1206 で、入力待ちの間出力できるように、デジタル画像 111 の  $i$  行  $j$  列の画素を記憶する。

【0098】ステップ 1207 で、デジタル画像 111 の帯域画素値を出力する。その後ステップ 1203 に戻る。

【0099】これにより、配列演算ユニット 100 から構成されるデータ処理装置 110 を用いて、画像記憶手段 12 はデジタル画像 111 を記憶することができる。

【0100】データ処理装置 110 によって実現される画像振動手段がデジタル画像 111 を振動させるために、二次元格子状に配列された配列演算ユニット 100 は同期して並列に動作する。格子上  $i$  行  $j$  列に配置された配列演算ユニット 100 を  $AOU_{ij}$  とすると、 $AOU_{ij}$  のアルゴリズムは図 9 のようになる。

【0101】ステップ 1301 で、 $AOU_{ij}$  を格子上の  $i$  行  $j$  列に配置する。これは論理的であれ物理的であれ、 $AOU_{ij}$  の近傍を決定するために必要である。

【0102】ステップ 1302 で、 $AOU_{ij}$  の近傍や変数の初期値を設定する。

【0103】ステップ 1303 で、順次入力されるデジタル画像 111 が無くなったかどうか判断する。もしデジタル画像 111 が無ければ (ステップ 1303: YES)、アルゴリズムを終了する。もしデジタル画像 111 があれば (ステップ 1303: NO)、ステップ 1304 に移行する。ただし特定の画像サイズのみに対して配列演算ユニット 100 を実装する場合には、無限ループにしても良い。

【0104】ステップ 1304 で、デジタル画像 111 の  $i$  行  $j$  列の画素を帯域数分入力する。このため  $AOU_{ij}$  は少なくとも帯域数分の画像データを記憶するメモリ 102 を必要とする。

【0105】ステップ 1305 で、関数  $S_{ijk}(x)$  に従いデジタル画像 111 の  $i$  行  $j$  列の画素を近傍画素の 1 つに移動させる。

【0106】ステップ 1306 で、デジタル画像 111 の帯域画素値を出力する。その後ステップ 1303 に戻る。

【0107】これにより、配列演算ユニット 100 から構成されるデータ処理装置 110 を用いて、画像振動手段はデジタル画像 111 を振動させることができる。

【0108】データ処理装置 110 によって実現されるエッジ情報生成手段 14 がデジタル画像 111 から粗エッジ情報画像 113 を生成するために、二次元格子状に配列された配列演算ユニット 100 は同期して並列に動作する。格子上  $i$  行  $j$  列に配置された配列演算ユニット 100 を  $AOU_{ij}$  とすると、エッジ情報生成手段 14

に対する  $AOU_{ij}$  のアルゴリズムは図 10 のようになる。

【0109】ステップ 1401 で、 $AOU_{ij}$  を格子上の  $i$  行  $j$  列に配置する。これは論理的であれ物理的であれ、 $AOU_{ij}$  の近傍を決定するために必要である。

【0110】ステップ 1402 で、 $AOU_{ij}$  の近傍や変数の初期値を設定する。近傍の設定においては、前記各関数で使う近傍サイズ  $q$  を個別に 4 から 8 に設定しても良いし、全部を 4 から 8 に統一しても良い。本発明のエッジ情報生成手段 14 が生成する粗エッジ情報 112 の正確さを上げるためには近傍サイズ  $q$  を全て 8 に設定することが望ましい。しかしながら粗エッジ情報 112 を生成するための計算時間の制約や、デジタル画像 111 の帯域数などにより、エッジ情報生成手段 14 は必要に応じて適宜近傍サイズを変えることで対処することができる。

【0111】ステップ 1403 で、デジタル画像 111 が終了したかどうか判断する。もしデジタル画像 111 が無ければ (ステップ 1403: YES)、アルゴリズムを終了する。もしデジタル画像 111 があれば (ステップ 1403: NO)、アルゴリズムを終了する。ただし特定の帯域数と画像サイズに対して配列演算ユニット 100 を実装する場合には、無限ループにしても良い。

【0112】ステップ 1404 で、デジタル画像 111 の  $i$  行  $j$  列の画素を帯域数分入力する。これは、 $AOU_{ij}$  がデジタル画像 111 の  $i$  行  $j$  列の画素を一括して処理するためである。このため  $AOU_{ij}$  は少なくとも帯域数分の画像データを記憶するメモリ 102 を必要とする。

【0113】ステップ 1405 で、 $AOU_{ij}$  が近傍の配列演算ユニット 100 と通信することにより、入力したデジタル画像 111 の各帯域画素値に対して関数  $S_{ijk}(x)$  に従い平滑化を行う。平滑化された帯域画素値は平滑化画像の帯域画素値として扱われる。ここで関数  $S_{ijk}(x)$  は必要に応じて数回繰り返しても良い。一般的な多帯域画像の場合、この回数は 2 回で十分である。

【0114】ステップ 1406 で、平滑化画像の各帯域画素値に対して関数  $L_{ijk}(x)$  に従い対数変換を行う。対数変換された帯域画素値は対数変換画像の帯域画素値として扱われる。

【0115】ステップ 1407 で、 $AOU_{ij}$  が近傍の配列演算ユニット 100 と通信することにより、対数変換画像の各帯域画素値に対して関数  $E_{ijk}(x)$  に従い鮮鋭化を行う。鮮鋭化された帯域画素値は鮮鋭化画像の帯域画素値として扱われる。

【0116】ステップ 1408 で、鮮鋭化画像の各帯域画素値に対して関数  $D_{ijk}(x, y)$  に従い 1 入力前鮮鋭化画像の各帯域画素値を引く。差分を計算された帯域画素値は時間差分画像の帯域画素値として扱われる。

【0117】ステップ1409で、1入力前鮮鋭化画像の各帯域画素値を鮮鋭化画像の対応する各帯域画素値で置き換える。

【0118】ステップ1410で、 $AOU_{ij}$  が近傍の配列演算ユニット100と通信することにより、時間差分画像の各帯域画素値に対してオペレータ $\nabla^2_{ijk}$  に従いラブラシアン $\nabla^2_{ijk}$  の計算を行う。ラブラシアンを計算された帯域画素値は時間差分ラブラシアン画像の帯域画素値として扱われる。

【0119】ステップ1411で、 $AOU_{ij}$  が近傍の配列演算ユニット100と通信することにより、時間差分ラブラシアン画像の各帯域画素値に対して関数 $B_{ij}(x)$  に従いゼロ点を抽出する。ゼロ点を抽出された帯域画素値は時間差分ゼロ点画像の帯域画素値として扱われる。

【0120】ステップ1412で、時間差分ラブラシアン画像の各帯域画素値に対して関数 $B_{ij}(x)$  に従い各帯域画素値のうち最大値を検出する。検出された最大値帯域画素値は最大値時間差分ゼロ点画像の帯域画素値として扱われる。なお便宜上帯域数は1である。

【0121】ステップ1413で、 $AOU_{ij}$  が近傍の配列演算ユニット100と通信することにより、鮮鋭化画像の各帯域画素値に対してオペレータ $\nabla^2_{ijk}$  に従いラブラシアン $\nabla^2_{ijk}$  の計算を行う。ラブラシアンを計算された帯域画素値はラブラシアン画像の帯域画素値として扱われる。

【0122】ステップ1414で、 $AOU_{ij}$  が近傍の配列演算ユニット100と通信することにより、ラブラシアン画像の各帯域画素値に対して関数 $B_{ij}(x)$  に従いゼロ点を抽出する。ゼロ点を抽出された帯域画素値はゼロ点画像の帯域画素値として扱われる。

【0123】ステップ1415で、ラブラシアン画像の各帯域画素値に対して関数 $B_{ij}(x)$  に従い各帯域画素値のうち最大値を検出する。検出された最大帯域画素値は最大値ゼロ点画像の帯域画素値として扱われる。なお便宜上帯域数は1である。

【0124】ステップ1416で、ラブラシアン画像の各帯域画素値と時間差分ラブラシアン画像の各帯域画素値に対して関数 $M_{ijk}(x, y)$  に従い各々の画像の同じ位置にある帯域画素値のうち最大値を検出する。検出された最大帯域画素値は混成ゼロ点画像の帯域画素値として扱われる。なお便宜上帯域数は1である。

【0125】ステップ1417で、 $AOU_{ij}$  が近傍の配列演算ユニット100と通信することにより、混成ゼロ点画像の帯域画素値に対して関数 $F_{ijk}(x)$  に従い孔を除去する。孔を除去された帯域画素値は孔除去混成ゼロ点画像の帯域画素値として扱われる。なお便宜上帯域数は1である。ここで関数 $F_{ijk}(x)$  は必要に応じて数回繰り返しても良い。一般的な多帯域画像の場合、この回数は1回で十分である。

【0126】ステップ1418で、 $AOU_{ij}$  が近傍の配列演算ユニット100と通信することにより、孔除去混成ゼロ点画像の帯域画素値に対して関数 $A_{ijk}(x)$  に従い孤立点および孤立孔を除去する。孤立点および孤立孔を除去された帯域画素値はノイズ除去混成ゼロ点画像の帯域画素値として扱われる。なお便宜上帯域数は1である。

【0127】ステップ1419で、ノイズ除去混成ゼロ点画像の帯域画素値に対して関数 $I_{ijk}(x)$  に従い0と1を反転させる。反転された帯域画素値は粗エッジ情報画像113の帯域画素値として扱われる。

【0128】ステップ1420で、粗エッジ情報画像113の帯域画素値を出力する。その後ステップ1403に戻る。

【0129】これにより、配列演算ユニット100から構成されるデータ処理装置110を用いて、エッジ情報生成手段14はデジタル画像111から粗エッジ情報画像113を生成することができる。

【0130】図11に示すように、データ処理装置110によって実現されるエッジ情報形成手段15が粗エッジ情報112から構成される粗エッジ情報画像113、及びデジタル画像111から、形成エッジ情報114から構成される形成エッジ情報画像115を生成するため、二次元格子状に配列された配列演算ユニット100は同期して並列に動作する。格子上1行1列に配置された配列演算ユニット100を $AOU_{ij}$  とすると、 $AOU_{ij}$  のアルゴリズムは図12のようになる。

【0131】ステップ1501で、 $AOU_{ij}$  を格子上の1行1列に配置する。これは論理的であれ物理的であれ、 $AOU_{ij}$  の近傍を決定するために必要である。

【0132】ステップ1502で、 $AOU_{ij}$  の近傍や変数の初期値を設定する。近傍の設定においては、前記各関数で使う近傍サイズ $q$ を個別に4か8に決めても良いし、全部を4か8に統一しても良い。本発明のエッジ情報形成手段15が形成した形成エッジ情報114の正確さを上げるためには近傍サイズ $q$ を全て8に設定することが望ましい。しかしながら粗エッジ情報112を形成するための計算時間の制約や、入力されるデジタル画像111の帯域数などにより、エッジ情報形成手段15は必要に応じて適宜近傍サイズを変えることで対処することができる。

【0133】ステップ1503で、順次入力されるデジタル画像111又は粗エッジ情報画像113が無くなくなったかどうか判断する。もしデジタル画像111若しくは粗エッジ情報画像113のいずれかが無ければ(ステップ1503: YES)、アルゴリズムを終了する。もしデジタル画像111若しくは粗エッジ情報画像113のいずれかがあれば(ステップ1503: NO)、ステップ1504に移行する。ただし特定の帯域数と画像サイズに対して配列演算ユニット100を実装する場合に

は、無限ループにしても良い。

【0134】ステップ1504で、デジタル画像111及び粗エッジ情報画像113の1行j列の画素を帯域数分入する。これは、 $AOU_{ij}$ がデジタル画像111及び粗エッジ情報画像113の1行j列の画素を一括して処理するためである。このため $AOU_{ij}$ は少なくとも帯域数分の画像データを記憶するメモリ102を必要とする。

【0135】ステップ1505で、デジタル画像111の1行j列の画素と粗エッジ情報画像113の1行j列の画素を分離する。これは、 $AOU_{ij}$ がデジタル画像111の1行j列の画素と粗エッジ情報画像113の1行j列の画素を各々独立した画像の画素として処理するためである。もしデジタル画像111の1行j列の画素と粗エッジ情報画像113の1行j列の画素が初めから分離されて入力されていれば、特に何もしない。

【0136】ステップ1506で、 $AOU_{ij}$ が近傍の配列演算ユニット100と通信することにより、入力したデジタル画像111の各帯域画素値に対して関数 $S_{ijk}(x)$ に従い平滑化を行う。平滑化された帯域画素値は平滑化画像の帯域画素値として扱われる。ここで関数 $S_{ijk}(x)$ は必要に応じて数回繰り返しても良い。一般的な多帯域画像の場合、この回数は2回で十分である。

【0137】ステップ1507で、平滑化画像の各帯域画素に対して関数 $L_{ijk}(x)$ に従い対数変換を行う。対数変換された帯域画素値は対数変換画像の帯域画素値として扱われる。

【0138】ステップ1508で、 $AOU_{ij}$ が近傍の配列演算ユニット100と通信することにより、対数変換画像の各帯域画素値に対して関数 $E_{ijk}(x)$ に従い鮮鋭化を行う。鮮鋭化された帯域画素値は鮮鋭化画像の帯域画素値として扱われる。

【0139】ステップ1509で、 $AOU_{ij}$ が近傍の配列演算ユニット100と通信することにより、鮮鋭化画像の各帯域画素値に対してオペレータ $\nabla^2_{ijk}x$ に従いラプラシアン計算を行う。ラプラシアンを計算された帯域画素値はラプラシアン画像の帯域画素値として扱われる。

【0140】ステップ1510で、 $AOU_{ij}$ が近傍の配列演算ユニット100と通信することにより、ラプラシアン画像の各帯域画素値に対して関数 $Z_{ijk}(x)$ に従いゼロ点を抽出する。ゼロ点を抽出された帯域画素値はゼロ点画像の帯域画素値として扱われる。

【0141】ステップ1511で、ゼロ点画像の各帯域画素値に対して関数 $B_{ijk}(x)$ に従い各帯域画素値のうち最大値を検出する。検出された最大帯域画素値は最大値ゼロ点画像の帯域画素値として扱われる。なお便宜上帯域数は1である。

【0142】ステップ1512で、最大値ゼロ点画像の

帯域画素値に対して関数 $I_{ijk}(x)$ に従い0と1を反転させる。反転された帯域画素値は基礎エッジ情報画像の帯域画素値として扱われる。

【0143】ステップ1513で、入力した粗エッジ情報画像113の帯域画素値は初め整形粗エッジ情報画像の帯域画素値として扱われ、 $AOU_{ij}$ が近傍の配列演算ユニット100と通信することにより、基礎エッジ情報画像の帯域画素値を用いて、整形粗エッジ情報画像の帯域画素値に対して関数 $Q_{ijk}(x, y)$ に従い整形を行う。整形された帯域画素値は再び整形粗エッジ情報画像の帯域画素値として扱われる。ここで関数 $Q_{ijk}(x, y)$ は本来整形粗エッジ情報画像の帯域画素値が変化しなくなるまで繰り返される。ただし計算時間の制約、入力される粗エッジ情報画像113の品質、形成される形成エッジ情報画像115に求められる品質などにより、整形処理は適当な繰り返し回数で計算を打ち切った方がよい。

【0144】ステップ1514で、 $AOU_{ij}$ が近傍の配列演算ユニット100と通信することにより、整形粗エッジ情報画像の帯域画素値に対して関数 $C_{ijk}(x)$ に従い線幅補完を行う。補完された帯域画素値は形成エッジ情報画像115の帯域画素値として扱われる。

【0145】ステップ1515で、形成エッジ情報画像115の帯域画素値を出力する。その後ステップ1503に戻る。

【0146】これにより、配列演算ユニット100から構成されるデータ処理装置110を用いて、エッジ情報形成手段115は粗エッジ情報画像113を形成エッジ情報画像115に形成することができる。

【0147】図13に示すように、データ処理装置110によって実現される領域正規化手段27が物体領域141を含む物体領域画像142、及び物体領域141と重なる分離物体領域143を含むデジタル画像111から正規化領域144を含む正規化画像145を生成するために、二次元格子状に配列された配列演算ユニット100は同期して並列に動作する。格子上1行j列に配置された配列演算ユニット100を $AOU_{ij}$ とすると、 $AOU_{ij}$ のアルゴリズムは図14のようになる。

【0148】ステップ2701で、 $AOU_{ij}$ を格子上の1行j列に配置する。これは論理的であれ物理的であれ、 $AOU_{ij}$ の近傍を決定するために必要である。

【0149】ステップ2702で、 $AOU_{ij}$ の近傍や変数の初期値を設定する。近傍の設定においては、前記各関数で使う近傍サイズqを個別に決めても良いし、全部を統一しても良い。本発明の領域正規化手段27が生成した正規化画像145の正確さ上げるためには近傍サイズqを全て大きな値に設定することが望ましい。しかしながら分離物体領域143を正規化するための計算時間の制約や、入力されるデジタル画像111のサイズ

などにより、領域正規化手段27は必要に応じて適宜近傍サイズを変えることで対処することができる。

【0150】ステップ2703で、順次入力される物体領域画像142又はデジタル画像111がなくなったかどうか判断する。もし物体領域画像142又はデジタル画像111が無ければ(ステップ2703: YES)、アルゴリズムを終了する。もし物体領域画像142又はデジタル画像111があれば(ステップ2703: NO)、ステップ2704に移行する。ただし特定の帯域数及び画像サイズのみに対して配列演算ユニット100を実装する場合には、無限ループにしても良い。

【0151】ステップ2704で、物体領域画像142のi行j列の画素を1帯域分と、デジタル画像111のi行j列の画素を帯域数分を入力する。これは、 $AOU_{ij}$  が物体領域画像142のi行j列の画素とデジタル画像111のi行j列の画素を一括して処理するためである。このため $AOU_{ij}$  は少なくともと総帯域数分の画像データを記憶するメモリ102を必要とする。

【0152】ステップ2705で、物体領域画像142のi行j列の画素とデジタル画像111のi行j列の画素を分離する。これは、 $AOU_{ij}$  が物体領域画像142のi行j列の画素とデジタル画像111のi行j列の画素を各々独立した画像の画素として処理するためである。もし物体領域画像142のi行j列の画素とデジタル画像111のi行j列の画素が初めから分離されて入力されていれば、特に何もしない。物体領域画像142及びデジタル画像111は各々更新物体領域画像及び更新デジタル画像にコピーされる。

【0153】ステップ2706で、 $AOU_{ij}$  が近傍の配列演算ユニット100と通信することにより、更新物体領域画像の各帯域画素値に対して関数 $R_{i,j,k}(x)$  に従い移動量を計算する。移動量と画像化した帯域画素値は移動量画像としての帯域画素値として扱われる。

【0154】ステップ2707で、 $AOU_{ij}$  が近傍の配列演算ユニット100と通信することにより、更新物体領域画像の各帯域画素値に対して関数 $H_{i,j,k}(x, y)$  に従い移動可能な移動先帯域画素値を見つけることができる。移動可能な移動先であるかどうかを表す値は移動可能画像の帯域画素値として扱われる。

【0155】ステップ2708で、 $AOU_{ij}$  が近傍の配列演算ユニット100と通信することにより、更新物体領域画像の各帯域画素値に対して関数 $U_{i,j,k}(x, y)$  に従い移動可能先に移動させる。移動した帯域画素値は新たに更新物体領域画像の帯域画素値として扱われる。

【0156】ステップ2709で、 $AOU_{ij}$  が近傍の配列演算ユニット100と通信することにより、更新デジタル画像の各帯域画素値に対して関数 $U_{i,j,k}(x, y)$  に従い移動可能先に移動させる。移動した帯域画素値は新たに更新デジタル画像の帯域画素値として扱われる。

る。

【0157】ステップ2710で、ステップ2706からステップ2709までの繰り返し回数を表す移動回数が指定回数に達したかどうか判断する。もし移動回数が指定回数に達していなければ(ステップ2710: NO)、ステップ2706に戻る。もし移動回数が指定回数に達していれば(ステップ2710: YES)、ステップ2711に移行する。なおこの指定回数はデジタル画像111のサイズやデジタル画像111の分離物体領域143のサイズ、さらには近傍のサイズqにより決定される。利用目的に応じて適切なパラメータを設定すれば、指定回数を大目に決定しても問題はないが、あまり指定回数を多くしすぎると、正規化に要する時間が長くなる。

【0158】ステップ2711で、 $AOU_{ij}$  が近傍の配列演算ユニット100と通信することにより、移動を完了した更新物体領域画像の各帯域画素値に対して関数 $V_{i,j,k}(x, y)$  に従い近傍の平均値で埋める。なおxとyは共に更新物体領域画像となる。平均値で埋められた帯域画素値は正規化された更新物体領域画像の帯域画素値として扱われる。

【0159】ステップ2712で、 $AOU_{ij}$  が近傍の配列演算ユニット100と通信することにより、移動を完了した更新デジタル画像の各帯域画素値に対して関数 $V_{i,j,k}(x, y)$  に従い近傍の平均値で埋める。なおxは更新デジタル画像となり、yは更新物体領域画像となる。平均値で埋められた帯域画素値は正規化された更新デジタル画像の帯域画素値として扱われる。

【0160】ステップ2713で、ステップ2711からステップ2712までの繰り返し回数を表す補間回数が指定回数に達したかどうか判断する。もし補間回数が指定回数に達していなければ(ステップ2713: NO)、ステップ2711に戻る。もし補間回数が指定回数に達していれば(ステップ2713: YES)、ステップ2714に移行する。一般的に補間回数は近傍サイズqの半程度程度の回数で十分である。

【0161】ステップ2714で、ステップ2706からステップ2713までの繰り返し回数を表す継続回数が指定回数に達したかどうか判断する。もし継続回数が指定回数に達していなければ(ステップ2714: NO)、ステップ2706に戻る。もし継続回数が指定回数に達していれば(ステップ2714: YES)、ステップ2715に移行する。なおこの指定回数はデジタル画像111のサイズやデジタル画像111の分離物体領域143のサイズ、さらには近傍のサイズqにより決定される。利用目的に応じて適切なパラメータを設定すれば、指定回数を大目に決定しても問題はないが、あまり指定回数を多くしすぎると、正規化に要する時間が長くなる。

【0162】ステップ2715で、更新デジタル画像の

帯域画素値を正規化画像145の帯域画素値として出力する。その後ステップ2703に戻る。

【0163】これにより、配列演算ユニット100から構成されるデータ処理装置110を用いて、領域正規化手段27が物体領域画像142及びデジタル画像111から正規化画像145を生成することができる。

【0164】図15に示すように、データ処理装置110によって実現される位置/大きさ検出手段17が粗エッジ情報112を画素とする粗エッジ情報画像113から重複情報131を画素とする重複情報画像132を生成するために、二次元格子状に配列された配列演算ユニット100は同期して並列に動作する。格子上一行j列に配置された配列演算ユニット100を $AOU_{1j}$ とすると、 $AOU_{1j}$ のアルゴリズムは図16のようになる。

【0165】ステップ1701で、 $AOU_{1j}$ を格子上の1行j列に配置する。これは論理的であれ物理的であれ、 $AOU_{1j}$ の近傍を決定するために必要である。

【0166】ステップ1702で、 $AOU_{1j}$ の近傍や変数の初期値を設定する。近傍の設定においては、前記各関数で使う近傍サイズqを個別に決めても良いし、全部を統一しても良い。本発明のデータ処理装置110が生成した重複情報画像132の正確さを上げるためには近傍サイズqを全て大きな値に設定することが望ましい。しかしながら物体の粗エッジ情報112の重心を計算するための計算時間の制約や、入力される粗エッジ情報画像113のサイズなどにより、位置/大きさ検出手段17は必要に応じて適宜近傍サイズを変えることで対処することができる。

【0167】ステップ1703で、順次入力される粗エッジ情報画像113が無くなったかどうか判断する。もし粗エッジ情報画像113が無ければ(ステップ1703: YES)、アルゴリズムを終了する。もし粗エッジ情報画像113があれば(ステップ1703: NO)、ステップ1704に移行する。ただし特定の画像サイズのみに対して配列演算ユニット100を実装する場合には、無限ループにしても良い。

【0168】ステップ1704で、粗エッジ情報画像113の1行j列の画素を1帯域分入力する。このため $AOU_{1j}$ は少なくとも1帯域分の画像データを記憶するメモリ102を必要とする。

【0169】ステップ1705で、粗エッジ情報画像113の粗エッジ情報112を重複情報画像132の重複情報131に変換する。重複情報131は1か0に相当する帯域画素値となる。

【0170】ステップ1706で、 $AOU_{1j}$ が近傍の配列演算ユニット100と通信することにより、重複情報画像132の各帯域画素値に対して関数 $\Delta$

$i, j, 1(x)$ に従い移動量を計算する。移動量を画像化した帯域画素値は移動量画像の帯域画素値として扱われ

る。

【0171】ステップ1707で、 $AOU_{1j}$ が近傍の配列演算ユニット100と通信することにより、重複情報画像132の各帯域画素値に対して関数 $\Delta$

$i, j, 1(x)$ に従い移動させる。移動した帯域画素値は新たに重複情報画像132の帯域画素値として扱われる。

【0172】ステップ1708で、ステップ1705からステップ1707までの繰り返し回数を表す移動回数が指定回数に達したかどうか判断する。もし移動回数が指定回数に達していなければ(ステップ1708: NO)、ステップ1705に戻る。もし移動回数が指定回数に達していれば(ステップ1708: YES)、ステップ1709に移行する。なおこの指定回数は形成エッジ情報画像115のサイズや形成エッジ情報114が表す物体のサイズ、さらには近傍のサイズqにより決定される。利用目的に応じて適切なパラメータを設定すれば、指定回数を大目に決定しても問題はないが、あまり指定回数を多くしすぎると、位置及び大きさの検出に要する時間が長くなる。

【0173】ステップ1709で、 $AOU_{1j}$ が近傍の配列演算ユニット100と通信することにより、重複情報画像132の各帯域画素値に対して関数 $\Delta$

$i, j, 1(x)$ に従い移動量を計算する。移動量を画像化した帯域画素値は移動量画像の帯域画素値として扱われる。

【0174】ステップ1710で、 $AOU_{1j}$ が近傍の配列演算ユニット100と通信することにより、重複情報画像132の各帯域画素値に対して関数 $\Delta$

$i, j, 1(x)$ に従い移動させる。移動した帯域画素値は新たに重複情報画像132の帯域画素値として扱われる。

【0175】ステップ1711で、重複情報画像132の帯域画素値を出力する。その後ステップ1703に戻る。

【0176】なお重複情報画像132の各重複情報131はその位置を中心とした周辺にある粗エッジ情報112の線数を表すので、結果的にその位置を中心とした物体の大きさを意味することになる。

【0177】これにより、配列演算ユニット100から構成されるデータ処理装置110を用いて、位置/大きさ検出手段17は粗エッジ情報画像113から重複情報画像132を生成することができる。

【0178】ここで図16のアルゴリズムにおいて粗エッジ情報112から構成される粗エッジ情報画像113の代りに物体領域141を表す物体領域画像142を用いると、図17に示すように、データ処理装置110によって実現される位置/大きさ検出手段17は物体領域141を表す物体領域画像142からも重複情報131を表す重複情報画像132を生成することができる。た



だし物体領域画像142を用いた場合、重複情報画像132の各重複情報131はその位置を中心とした物体領域141の画素の総数を表すので、結果的にその位置を中心とした物体の面積を意味することになる。したがって重複情報画像132から物体の大きさを求める場合には重複情報131の平方根を取るなど注意を要する。

【0179】データ処理装置110によって実現される画像保持手段38がデジタル画像111、粗エッジ情報画像113、形成エッジ情報画像115、重複情報画像132、物体領域画像142、正規化画像145、及びマッチング結果画像147、を保持及び出力するために、二次元格子状に配列された配列演算ユニット100は同期して並列に動作する。格子上*i*行*j*列に配置された配列演算ユニット100を $AOU_{ij}$ とすると、デジタル画像111に対する $AOU_{ij}$ のアルゴリズムは図18ようになる。

【0180】ステップ3801で、 $AOU_{ij}$ を格子上の*i*行*j*列に配置する。これは論理的であれ物理的であれ、 $AOU_{ij}$ の近傍を決定するために必要である。

【0181】ステップ3802で、 $AOU_{ij}$ の近傍や変数の初期値を設定する。

【0182】ステップ3803で、順次入力されるデジタル画像111が無くなったかどうか判断する。もしデジタル画像111が無ければ(ステップ3803: YES)、アルゴリズムを終了する。もしデジタル画像111があれば(ステップ3803: NO)、ステップ3804に移行する。ただし特定の画像サイズのみに対して配列演算ユニット100を実装する場合には、無限ループにしても良い。

【0183】ステップ3804で、デジタル画像111の*i*行*j*列の画素を帯域数分入力する。このため $AOU_{ij}$ は少なくとも帯域数分の画像データを記憶するメモリ102を必要とする。

【0184】ステップ3805で、出力先の装置が必要とすればデジタル画像111のフォーマットを変換する。特にデジタル画像111の帯域数を1にしたり、前

入力データから構成される入力画像の帯域数が4以上の場合にデジタル画像111の帯域数を3にして、アナログ信号を生成しやすくする場合に便利である。さもなくば何もしない。

【0185】ステップ3806で、処理速度の異なる出力先の装置に画像データを確実に送信できるように、デジタル画像111の*i*行*j*列の画素を記憶する。

【0186】ステップ3807で、デジタル画像111の帯域画素値を出力する。その後ステップ3803に戻る。

【0187】これにより、配列演算ユニット100から構成されるデータ処理装置110を用いて、画像保持手段38がデジタル画像111を出力先の装置に出力することができる。

【0188】ここまではデータ処理装置110が1個又は2個の画像を入力して近傍処理のみからなる画像処理を行うような幾つかの手段について説明してきた。しかしながらパターンマッチング手段29は非常に多数の画像を用いなければならない。そこで以下ではパターンマッチング手段29で用いる近傍処理を示しながら、データ処理装置110によってパターンマッチング手段29を実現する方法について説明する。

【0189】まず任意の画像を $x$ とし、 $n$ 個のテンプレート画像146を $y^1, y^2, \dots, y^h, \dots, y^n$ とする。ただし、全てのテンプレート画像146の帯域数は画像 $x$ の帯域数と同じである。自然数 $g$ を用いると、マッチング結果画像147の*i*行*j*列のマッチング結果 $\delta_{ij1}$ は、数式37に従って画像 $x$ 及びテンプレート画像146の*i*行*j*列の画素を比較し、画像 $x$ の画素に最も似ている画素を有するテンプレート画像146の番号を指し示す。なおマッチング結果画像147は単帯域画像となるので、便宜上帯域数1の画像として取り扱うことにする。したがってマッチング結果 $\delta_{ij1}$ の第3添字は1となっている。

【0190】

【数37】

$$\delta_{ij1} = \begin{cases} g & \text{if } \sum_k (x_{ijk} - y_{ijk}^g)^2 = \min_{1 \leq k \leq n} \sum_k (x_{ijk} - y_{ijk}^k)^2 \\ & \text{for } 1 \leq g \leq n \text{ and only one of } g, \\ 0 & \text{otherwise.} \end{cases}$$

【0191】ここで数式37に従って生成されたマッチング結果 $\delta_{ij1}$ はマッチング結果画像147全体において必ずしも統一されていない。テンプレート画像146が多数ある場合、マッチング結果画像147はむしろモザイク状になる可能性が高い。そこでデータ処理装置110がマッチング結果 $\delta_{ij1}$ とその $q$ 近傍内のマッチング結果に対するヒストグラムを計算し、マッチング結果 $\delta_{ij1}$ を収束する方法を以下に示す。

【0192】任意の単帯域画像 $x$ がマッチング結果画像

147である場合、自然数 $g$ 、実数 $u$ と $v$ を用いると、マッチング結果画像147は数式38及び39に従って更新される。なおマッチング結果画像147は単帯域画像となるので、便宜上帯域数1の画像として取り扱うことにする。したがって関数 $\psi_{ij1}(x)$ の第3添字は1となっている。

【0193】

【数38】

$$\Psi_{ij1}(\underline{x}) = \begin{cases} g & \text{if } eq(g, x_{ij1}) + \sum_{p(l,m,1) \in P_{ij1}(g)} eq(g, x_{lm1}) = \\ & \max_{1 \leq h \leq n} \{eq(h, x_{ij1}) + \sum_{p(l,m,1) \in P_{ij1}(h)} eq(h, x_{lm1})\} \\ & \text{and } 2\{eq(g, x_{ij1}) + \sum_{p(l,m,1) \in P_{ij1}(g)} eq(g, x_{lm1})\} \geq N_{ij1} \\ & \text{for } 1 \leq g \leq n \text{ and only one of } g, \\ 0 & \text{otherwise.} \end{cases}$$

【0194】

【数39】

$$eq(u, v) = \begin{cases} 1 & \text{if } u = v, \\ 0 & \text{otherwise.} \end{cases}$$

【0195】マッチング結果画像147が変化しなくなるまでデータ処理装置110が数式38及び39を繰り返し計算することにより、マッチング結果画像147全体のマッチング結果を収収することができる。このとき画像xとテンプレート画像146の組み合わせにより、マッチング結果は次のように収収する。もし画像xの約半分の画素が特定のテンプレート画像146の画素に最も類似していれば、マッチング結果画像147の殆どこのマッチング結果はこの特定のテンプレート画像146の番号に収収する。しかしながら画像xの幾つかの画素の塊が幾つかの異なるテンプレート画像146の画素の塊と類似していれば、マッチング結果画像147には0で囲まれた幾つかのテンプレート画像146の番号の塊ができる。さらに画像xがテンプレート画像146の集合と相関がなければ、マッチング結果画像147のマッチング結果は殆んど0となる。したがってデータ処理装置110によって実現されるパターンマッチング手段29は、画像xに最も似ているテンプレート画像146を特定することは難しいが、テンプレート画像146の中から幾つかの似ているテンプレート画像146を選択することができると思われる。

【0196】なおタッチセンサにパターンマッチング手段29を実装した場合、タッチセンサに接続される後段の処理装置においてパターンマッチング手段29のマッチング結果画像147から移動物体及び静止物体の種別を生成する過程では、パターンマッチング手段29によって生成されたマッチング結果画像147が列挙するテンプレート画像146の類似候補の中から多数決などを行うことにより、最も有力な候補1つを選択するだけで良い。

【0197】図19に示すように、データ処理装置110によって実現されるパターンマッチングが、テンプレート画像146のうち正規化画像145に最も似ている画像の番号を示すマッチング結果から構成されるマッチング結果画像147を生成するために、二次元格子状に配列された配列演算ユニット100は同期して並列に動作する。格子上i行j列に配置された配列演算ユニット100を $AOU_{ij}$ とすると、 $AOU_{ij}$ のアルゴリズムは図20のようになる。

【0198】ステップ2901で、 $AOU_{ij}$ を格子上のi行j列に配置する。これは論理的であれ物理的であれ、 $AOU_{ij}$ の近傍を決定するために必要である。

【0199】ステップ2902で、 $AOU_{ij}$ の近傍や変数の初期値を設定する。近傍の設定においては、前記各関数で使う近傍サイズqを個別に決めても良いし、全部を統一しても良い。本発明のデータ処理装置110が生成したマッチング結果画像147の正確さを上げるためには近傍サイズqを全て大きな値に設定することが望ましい。しかしながらマッチング結果を更新するための計算時間の制約や、入力される正規化画像145のサイズなどにより、パターンマッチングは必要に応じて適宜近傍サイズを変えることで対応することができる。

【0200】ステップ2903で、順次入力されるテンプレート画像146が無くなったかどうか判断する。もしテンプレート画像146が無ければ（ステップ2903：YES）、ステップ2905に移行する。もしテンプレート画像146があれば（ステップ2903：NO）、ステップ2904に移行する。

【0201】ステップ2904で、テンプレート画像146のi行j列の画素を帯域数分入力する。このため $AOU_{ij}$ は少なくとも帯域数とテンプレート画像146の数を掛けた分の画像データを記憶するメモリ102を必要とする。その後ステップ2903に戻る。

【0202】ステップ2905で、順次入力される正規化画像145が無くなったかどうか判断する。もし正規化画像145が無ければ（ステップ2905：YES）、アルゴリズムを終了する。もし正規化画像145があれば（ステップ2905：NO）、ステップ2906に移行する。ただし特定の画像サイズのみに対して配列演算ユニット100を実装する場合には、無限ループにしても良い。

【0203】ステップ2906で、正規化画像145のi行j列の画素を帯域数分入力する。このため $AOU_{ij}$ は少なくとも帯域数分の画像データを記憶するメモリ102を必要とする。

【0204】ステップ2907で、正規化画像145とテンプレート画像146からマッチング結果画像147のマッチング結果 $\delta_{ij1}$ を計算する。マッチング結果は正規化画像145に最も近いテンプレート画像146の番号を表す帯域画素値となる。

【0205】ステップ2908で、 $AOU_{ij}$ が近傍の配列演算ユニット100と通信することにより、マッチング結果画像147の各帯域画素値に対して関数 $\Psi$

1.3.1 (x) に従いマッチング結果を更新する。更新された帯域画素値は再びマッチング結果画像 147 の帯域画素値として扱われる。ここで関数  $\Psi_{1.3.1}(x)$  は本来マッチング結果画像 147 の帯域画素値が変化しなくなるまで繰り返される。ただし計算時間の制約、入力される正規化画像 145 の品質、更新されたマッチング結果画像 147 に求められる品質などにより、更新処理は適当な繰り返し回数で計算を打ち切った方がよい。

【0206】ステップ 2909 で、マッチング結果画像 147 の帯域画素値を出力する。その後ステップ 2910 に戻る。

【0207】これにより、配列演算ユニット 100 から構成されるデータ処理装置 110 を用いて、パターンマッチング手段 29 は正規化画像 145 からマッチング結果画像 147 を生成することができる。

【0208】ここまでは配列演算ユニット 100 から構成されるデータ処理装置 110 を用いて、近傍処理のみからなる画像処理を行う方法について説明してきた。以下では配列演算ユニット 100 から構成されるデータ処理装置 110 を用いて、近傍処理のみで物体/背景分離手段 16 を行う方法について説明する。

【0209】まず非線形振動子は一般に引き込み現象を起こす。この引き込み現象とは、リミットサイクルやアトラクタなどのような周期的挙動において、異なる周期を持つ非線形振動子が相互作用して簡単な定数比の周期で振動するよう制約される現象である。このとき 1 つの非線形振動子の振動を変化させると他の非線形振動子の振動も合わせて変化する。これらの非線形振動子は

$$\Omega_{ij}(q) = \begin{cases} \{\omega_{i+1,j}, \omega_{i,j+1}, \omega_{i-1,j}, \omega_{i,j-1}\} & \text{if } q = 4, \\ \{\omega_{i,m} \mid i-r \leq l \leq i+r, j-r \leq m \leq j+r, \omega_{i,m} \neq \omega_{i,j}\} & \text{if } q = (2r+1)^2 - 1. \end{cases}$$

【0212】次に、非線形振動子は  $q_{\mu}$  近傍に含まれる近傍集合  $\Omega_{i,j}(q_{\mu})$  における非線形振動子との間で数式 4.1 に従い計算される結合値  $\tau_{ijkl}$  によって結合される。なお対数表を用いない場合には数式 4.2 による

$$\tau_{ijkl} = \mu \operatorname{sinc}\left(\frac{(i-k)^2 + (j-l)^2}{\nu^2}\right) \quad \text{for } \forall \omega_{kl} \in \Omega_{ij}(q_{\mu})$$

【0214】

$$\operatorname{sinc}(x) \approx \begin{cases} 1 - 2|x|^2 + |x|^3 & \text{if } 0 \leq |x| < 1, \\ 4 - 8|x| + 5|x|^2 - |x|^3 & \text{if } 1 \leq |x| < 2, \\ 0 & \text{otherwise.} \end{cases}$$

【0215】非線形振動子ネットワークの全ての非線形振動子が完全に同位相で同期した場合、プロセッサ 101 で計算する限り、非線形振動子  $\omega_{i,j}$  は永久に同位相のまま動作し続けてしまう。そこで外乱  $\rho_{i,j}$  を与え

同期している。しかも非線形振動子の相互作用を調整することにより、お互いの振動の位相差を極力小さくさせたり大きくさせたりできる。そこでこの相互作用を操作すると、非線形振動子の一群を、異なる位相を持つ複数のグループに分割することができる。物体/背景分離手段 16 はこのような非線形振動子の引き込み現象を利用して、エッジ情報画像中のエッジ情報を境界とするように物体と背景を分離して、物体領域 141 を表す物体領域画像 142 を生成する。なお、ここでは非線形振動子としてファン・デル・ポールを用いた場合を例に説明する。

【0210】まず、二次元格子状に配列した非線形振動子から構成される非線形振動子ネットワークにおいて、 $i$  行  $j$  列にある非線形振動子を  $\omega_{i,j}$  とすると、非線形振動子  $\omega_{i,j}$  の  $q$  近傍にある非線形振動子の集合  $\Omega_{i,j}(q)$  は数式 4.0 によって表される。ただし  $q$  は 4、8、24、48、80、120、 $(2r+1)^2 - 1$  と続く数列であり、 $r$  は自然数である。なおネットワークサイズをはみ出した非線形振動子が近傍集合  $\Omega_{i,j}(q)$  に含まれる場合には、非線形振動子  $\omega_{i,j}$  を代用するものとする。これにより辺縁処理は自動的に行われる。したがって近傍集合  $\Omega_{i,j}(q)$  の要素の数は常に  $q$  となる。なおこのことから判る通り、非線形振動子ネットワークは単帯域画像と同じ扱いになる。表現を簡単にするため、非線形振動子ネットワークでは添字は幅方向と高さ方向の 2 つのみを使う。

【0211】

【数 4.0】

【数 4.0】

【数 4.0】

【数 4.0】

【数 4.0】

【数 4.0】

【数 4.0】

【数 4.0】

【数 4.0】

【数 4.0】

【数 4.0】

【数 4.0】

【数 4.0】

【数 4.0】

【数 4.0】

【数 4.0】

【数 4.0】

【数 4.0】

【数 4.0】

【数 4.0】

【数 4.0】

【数 4.0】

【数 4.0】

【数 4.0】

【数 4.0】

【数 4.0】

【数 4.0】

近似も可能である。また  $\mu$ 、 $\nu$  は適当な正の定数である。

【0213】

【数 4.1】

【数 4.1】

【数 4.1】

【数 4.1】

【数 4.1】

【数 4.1】

【数 4.1】

【数 4.1】

【数 4.1】

【数 4.1】

【数 4.1】

【数 4.1】

【数 4.1】

【数 4.1】

【数 4.1】

【数 4.1】

ばこのような状態を回避することができる。外乱としては疑似乱数を用いることもできるが、数式 4.3 のような簡単な式で求めても十分である。なお  $\rho_{i,j}$  はエッジ情報画像の  $i$  行  $j$  列のエッジ情報の有無を表す。エッジ情

報があれば1とし、なければ0とする。また $\kappa$ は適当な正の定数である。

【0216】

【数43】

$$\rho_{ij} = \kappa \zeta_{ij}$$

【0217】非線形振動子 $\omega_{ij}$ が近傍集合 $\Omega_{ij}$  ( $q_a$ )の非線形振動子 $\omega_{k1}$ と同期するために、数式44に従い近傍入力合計 $\sigma_{ij}$ を計算する。

【0218】

【数44】

$$\sigma_{ij} = \sum_{\omega_{kl} \in \Omega_{ij}(q_a)} \tau_{ijkl}(1 - \zeta_{kl}) \xi_{kl}(\psi_{kl} - \xi_{ij})$$

【0219】ファン・デル・ポール非線形振動子 $\omega_{ij}$ を構成する2つのパラメータ $\phi_{ij}$ と $\psi_{ij}$ は数式45及び46に従って計算される。なお $\tau$ 、 $\epsilon$ は適当な正の定数である。

【0220】

【数45】

$$\frac{d\phi_{ij}}{dt} = \psi_{ij}$$

【0221】

【数46】

$$\frac{d\psi_{ij}}{dt} = -\gamma\phi_{ij} - \epsilon(1 - \phi_{ij}^2)\psi_{ij} + \sigma_{ij} + \rho_{ij}$$

【0222】非線形振動子を物体領域141と背景領域に分離するためには全ての非線形振動子の位相のずれを計算する必要があるが、単純に物体領域141と背景領域の2つに分離するだけであるので、パラメータ $\psi_{ij}$ が小さい値0以上かつ未満かつ位相ずれを計算する。物体領域141と背景領域を分離した結果を出力する出力 $\lambda_{ij}$ は数式47によって求められる。なお $\theta$ は適当な正の定数である。

【0223】

【数47】

$$\lambda_{ij} = \begin{cases} 1 & \text{if } \psi_{ij} \geq \theta, \\ 0 & \text{otherwise.} \end{cases}$$

【0224】エッジ情報が物体と背景を分離するのに十分であった場合にはエッジ情報を補間しなくてはならない。そのために非線形振動子 $\omega_{ij}$ の $q_b$ 近傍にある非線形振動子の集合 $\Omega_{ij}$  ( $q_b$ )中でいくつかの非線形振動子が位相ずれを起こしているか求める必要がある。そこで数式48によって輪郭パラメータ $\eta_{ij}$ を計算する。

【0225】

【数48】

$$\eta_{ij} = \sum_{\omega_{kl} \in \Omega_{ij}(q_b)} \lambda_{ij} \lambda_{kl} + (\lambda_{ij})^2$$

【0226】この結果を基にエッジ情報の補間割合を示す境界パラメータ $\xi_{ij}$ を数式49によって計算する。

なお $\alpha$ 、 $\beta$ 、 $\eta_{min}$ 、 $\eta_{max}$ は適当な正の定数である。

【0227】

【数49】

$$\frac{d\xi_{ij}}{dt} = \begin{cases} -\alpha\xi_{ij} & \text{if } \eta_{min} \leq \eta_{ij} \leq \eta_{max}, \\ \beta(1 - \xi_{ij}) & \text{otherwise.} \end{cases}$$

【0228】ここでは非線形振動子としてファン・デル・ポールの場合を説明したが、この他にプラセレータのようなリミットサイクルで安定する非線形振動子や、ローレンツアトラクタやレスラー方程式のアトラクタを発生するカオス振動子など、引き込み現象を起こす任意の非線形振動子でも動作可能である。その場合はパラメータ $\phi_{ij}$ と $\psi_{ij}$ を各々の非線形振動子のパラメータで置き換えるなり追加すれば良い。その際に適当なパラメータに近傍入力合計 $\sigma_{ij}$ と外乱 $\rho_{ij}$ を加えるだけである。但し、カオス振動子の場合には特に外乱 $\rho_{ij}$ を必要としない。

【0229】数式40から数式49までを用いることにより、物体/背景分離手段16を実装することができるデータ処理装置110の全ての配列演算ユニット100のアルゴリズムを記述することができる。以下では、データ処理装置110中の任意の配列演算ユニット100のアルゴリズムを用いて、物体/背景分離手段16について説明する。

【0230】図21に示すように、データ処理装置110によって実現される物体/背景分離手段16が、三角形のエッジ情報151を用いて三角形の内側領域152と三角形の外側領域153に分離するために、二次元格子状に配列された配列演算ユニット100は同期して並列に動作する。格子上1行j列に配置された配列演算ユニット100を $AOU_{ij}$ とすると、 $AOU_{ij}$ のアルゴリズムは図22のようになる。

【0231】ステップ1601で、 $AOU_{ij}$ を格子上の1行j列に配置する。

【0232】ステップ1602で、数式41及び42に基づいて近傍同士 $\omega_{ij}$ と $\omega_{k1}$ を結合値 $\sigma_{ijkl}$ と接続する。

【0233】ステップ1603で、非線形振動子のパラメータ $\phi_{ij}$ と $\psi_{ij}$ に適当な初期値を設定する。

【0234】ステップ1604で、順次入力される形成エッジ情報画像115が無くなったかどうか判断する。もし形成エッジ情報画像115が無ければ(ステップ1604: YES)、アルゴリズムを終了する。もし形成エッジ情報画像115があれば(ステップ1604: NO)、ステップ1605に移行する。ただし特定の帯域数及び画像サイズのみに対して配列演算ユニット100を実装する場合には、無限ループにしても良い。

【0235】ステップ1605で、形成エッジ情報114の $\xi_{ij}$ を入力する。

【0236】ステップ1606で、直前に入力した形成エッジ情報114の $\varepsilon_{ij}$ から数式43に従って外乱 $\rho_{ij}$ を計算する。

【0237】ステップ1607で、近傍集合 $\Omega_{ij}$  ( $q_a$ ) 中の非線形振動子 $\omega_{k1}$ がある配列演算ユニット100の $AOU_{k1}$ から $\lambda_{k1}$ 、 $\varepsilon_{k1}$ 、 $\psi_{k1}$ を入力して、合計値 $\sigma_{ij}$ を数式44に従って計算する。

【0238】ステップ1608で、非線形振動子のパラメータ $\phi_{ij}$ 、 $\psi_{ij}$ を数式45及び46に従って計算する。即ち、これらの数式に示す微分方程式をルンゲ・クッタ法で解く。

【0239】ステップ1609で、非線形振動子の出力 $\lambda_{ij}$ を数式47に従って計算する。ここで、 $\psi_{ij} \geq \theta$ であれば $\lambda_{ij} = 1$ とし、それ以外であれば $\lambda_{ij} = 0$ とする。

【0240】ステップ1610で、近傍集合 $\Omega_{ij}$  ( $q_b$ ) 中の非線形振動子 $\omega_{k1}$ がある配列演算ユニット100の $AOU_{k1}$ から $\lambda_{k1}$ を入力して、輪郭パラメータ $\kappa_{ij}$ を数式48に従って計算する。

【0241】ステップ1611で、境界パラメータ $\xi_{ij}$ を数式49に従って計算する。即ち、この数式に示す微分方程式を差分法若しくはルンゲ・クッタ法で解く。

【0242】ステップ1612で、ステップ1606からステップ1611までの繰り返し回数を表す分離回数が指定回数に達したかどうか判断する。もし分離回数が指定回数に達していなければ(ステップ1612: NO)、ステップ1606に戻る。もし分離回数が指定回数に達していれば(ステップ1612: YES)、ステップ1613に移行する。

【0243】ステップ1613で、物体領域画像142の帯域画素値となる非線形振動子の出力 $\lambda_{ij}$ を出力する。その後ステップ1604に戻る。

【0244】なおステップ1612での分離回数を求めるには、次のような方法を用いることができる。物体/背景分離手段16では、画像サイズが一定であれば非線形振動子の初期状態に関わらずおおよそ全ての形成エッジ情報114においてある一定時間で分離が終了するので、事前にこの時間を計っておいてステップ1606からステップ1611までの繰り返し回数を求めておけば良い。これは非線形振動子の初期状態が一定の範囲内であれば、引き込み現象により非線形振動子が同期するまでの時間はあまり大差がないからである。

【0245】このように非線形振動子を計算するだけで、三角形のエッジ情報151を用いて三角形の内側領域152と三角形の外側領域153を分離することができるのは、非線形振動子の性質である引き込み現象を利用しているからである。つまり、2つの非線形振動子を正の結合値で結合した場合は同位相になろうとし、負の結合値で結合した場合は位相差が極力大きくなろうとす

る。この性質を用いると、二次元格子状に並んだ非線形振動子を近傍同位相の結合値で結合することで、直接結合していない非線形振動子同士が同位相になる。さらに形成エッジ情報114を挟む画素の場所にある非線形振動子同士を負の結合値で結合すると、エッジ情報の両側がお互いに位相逆極力ずらし合う。このようにすることで、全ての非線形振動子を結合することなく三角形のエッジ情報151の内側と外側とで各々異なる非線形集合ができる。したがって物体/背景分離手段16は図21のような三角形の内側領域152と三角形の外側領域153に分離する。このとき三角形の内側領域152と三角形の外側領域153の位相差は90度を越えて可能な限り180度に近づき、三角形と背景領域が分離できる。

【0246】ここで重要なことは、本実施形態では、形成エッジ情報114が得られる度に次に示すような方法で結合値を擬似的に変更していることである。まず数式41及び42で定めたとおり、非線形振動子 $\omega_{k1}$ を非線形振動子 $\omega_{ij}$ に結合するための結合値を $\sigma_{ijk1}$ とする(ステップ1602参照)。形成エッジ情報114のうち $\varepsilon_{ij}$ と $\varepsilon_{k1}$ は共に、エッジがある場合には1、ない場合には0である。形成エッジ情報114のうち $\varepsilon_{ij}$ と $\varepsilon_{k1}$ を入力したら(ステップ1605参照)、配列演算ユニット100の $AOU_{k1}$ から $AOU_{ij}$ に形成エッジ情報114と $\varepsilon_{k1}$ が転送され、 $AOU_{ij}$ では結合値 $\sigma_{ijk1}(1 - \varepsilon_{k1})$ を計算して結合値 $\sigma_{ijk1}$ の代用とする(ステップ1607参照)。この代用された結合値 $\sigma_{ijk1}(1 - \varepsilon_{k1})$ に対して境界パラメータ $\xi_{ij}$ が0から1までの倍率として作用する(ステップ1607参照)。

【0247】図23に示す通り、形成エッジ情報114が破線状態の三角形のエッジ情報154となった場合には破線の補間をする必要がある。まず最初に破線状態の三角形のエッジ情報154を用いてシステムを動作させる(ステップ1605参照)と、破線状態の三角形のエッジ情報154の内側と外側で位相差がおおよそ90度を越えるようになるが、三角形の内側と外側の境界部分は不明確である。そこで各 $AOU_{ij}$ が非線形振動子の出力 $\lambda_{ij}$ を計算する(ステップ1609参照)。この出力 $\lambda_{ij}$ が1の場合、近傍の非線形振動子のうち $\lambda_{k1}$ が1である非線形振動子を $\omega_{k1}$ とすると、パラメータ $\psi_{ij}$ と $\psi_{k1}$ が共に $\theta$ 以上となる。つまり $\lambda_{ij}$ と $\lambda_{k1}$ はおおよそ同位相であり、 $\theta$ が正値であれば並進でも位相差が90度を越えることはない。この位相差の最大値は $\theta$ の値によって決まり、 $\lambda_{ij}$ と $\lambda_{k1}$ が共に1となる範囲で $\theta$ を大きくしていくと、この位相差は0度に近づいていく。したがって $\lambda_{ij}$ と $\lambda_{k1}$ を用いると、近傍の非線形振動子うちおおよそ同位相であるものの数を表す輪郭パラメータ $\kappa_{ij}$ は数式48に従って計算される(ステップ1610参照)。続いてこの輪郭パラメー

タ $\eta_{ij}$ が近傍全体のうち、およそ半分であれば結合値の倍率である境界パラメータ $\epsilon_{ij}$ を数式49に従って減少させ、それ以外であれば数式49に従って増加させる(ステップ1611参照)。例えば、8近傍の場合は3から5の間であれば数式49に従って境界パラメータを減少させるとよい。この過程を繰り返し動作させ続けると、図23に示す破線状態の三角形のエッジ情報154が与えられた場合、破線三角形の内側領域155と破線三角形の外側領域156に分離される。

【0248】図24に示す通り、2つの三角形が重なりあっている場合は、前方の三角形のエッジ情報157と後方の三角形のエッジ情報158が得られる。このとき前方三角形の内側領域159と後方三角形の内側領域160と二重三角形の背景領域161の3つの領域の非線形振動子の位相がお互いにずれることにより、3つの領域に分離される。また図25に示す通り、2つの重なった円形のエッジ情報162が破線であっても、前方円形の内側領域163と後方円形の内側領域164と二重円の背景領域165の3つに分離される。

【0249】これにより、配列演算ユニット100から構成されるデータ処理装置110を用いて、物体/背景分離手段16は形成エッジ情報画像115の形成エッジ情報114を境界として物体領域141と背景を分離することができる。

【0250】そこでデジタル技術を用いて画像記憶手段12、画像振動手段、エッジ情報生成手段14、エッジ情報形成手段15、物体/背景分離手段16、領域正規化手段27、位置/大きさ検出手段17、パターンマッチング手段29、及び画像保持手段38、を実装するために、配列演算ユニット100はデータ処理装置110中で図7のように二次元格字状に配列され、さらに配列演算ユニット100はデータ処理装置110中の隣接する配列演算ユニット100だけと相互に通信できるように配線される。つまり4近傍同士が直接配線されることになる。これにより8近傍同士を配線する場合に比べて、少ない電子部品と配線量で、同程度に高速に動作し、しかも将来近傍サイズを拡張する場合にも簡単に拡張性を有することができる。

【0251】配列演算ユニット100は図26に示す通り、画像処理における数式を計算するためのプロセッサ(PROCESSOR)101と、数式で使われる全てのパラメータ、定数、関数及びオペレータを記憶するためのメモリ(MEMORY)102と、近傍の配列演算ユニット100と通信するためのコントローラ(CONTROLLER)103から構成され、プロセッサ101及びコントローラ103はクロック信号(CLOCK)を入力し、プロセッサ101はメモリ102及びコントローラ103に対して読み込み(READ)及び書き込み(WRITE)を送信することにより通信を制御する。プロセッサ101はアドレスバス51で指定したアドレス(ADDRESS)によりメモリ102

及びコントローラ103の任意のメモリ素子及びレジスタを選択することができる。またプロセッサ101はデータバス52を介してメモリ102及びコントローラ103と双方向に通信可能に接続され、アドレスバス51で指定された任意のメモリ素子及びレジスタのデータ(DATA)にアクセスすることができる。配列演算ユニット100が1つ以上の入力画像から構成される前入力データ群(FRONT INPUT DATA SET)を入力すると、コントローラ103は前入力データ群をメモリ102に記憶させる。またコントローラ103は、関数により作成されたメモリ102中の計算データを隣接する配列演算ユニット100に送信すると共に、隣接する配列演算ユニット100から受信した計算データをメモリ102に記憶させ、さらに必要ならば、入力した以外の配列演算ユニット100に転送する。最終的にコントローラ103は、出力画像の画像データを結果データ(RESULT DATA)として出力する。

【0252】このように各配列演算ユニット100にコントローラ103を搭載する理由は、配列演算ユニット100同士が通信している間にプロセッサ101が動作できるので、プロセッサ101は通信による待ち時間中にも計算することができて高速処理が実現できると、近傍の配列演算ユニット100の数を変化させてもハードウェアを変更する必要があるからと、コントローラ103が画像の辺縁処理、つまり画像中の線の画像に対する例外処理を自動的に行えるので、プロセッサ101のプログラムは辺縁処理をする必要がなくなり極めて単純なからと、である。

【0253】プロセッサ101とメモリ102は汎用的なデジタル回路を用いることができる。コントローラ103の具体的な回路図は図27に示す通りである。アドレスバッファ(ADDRESS BUFFER)53はアドレスバス(ADDRESS BUS)51を介してプロセッサ101からアドレス(ADDRESS)を受取り、アドレスデコーダ(ADDRESS DECODER)54によって各レジスタ及びその他の電子回路ブロックを選択する。データバッファ(DATA BUFFER)55はデータバス(DATA BUS)52を介してプロセッサ101からデータ(DATA)を受取り、アドレスデコーダ54で選択されたレジスタと内部データバス56を介して排他的に通信する。通信方向は読み込み(READ)によって指定される。コントローラ103からプロセッサ101に向けてデータが送信される際、読み込みはアクティブになる。プロセッサ101からコントローラ103に向けてデータが送信される際、書き込み(WRITE)はアクティブになる。アドレスがフラグレジスタ(FLAG REGISTER)57を指定した場合、データはフラグレジスタ57に記憶され、フラグデコーダ(FLAG DECODE)58によってデコードされ、クロック信号(CLOCK)を用いて、複数信号(SIGNALS)として隣接する配列演算ユニット100に送信される。複数信号はフラグエン

コード (FLAG ENCODER) 59 によって受信され、解析された後にステータスレジスタ (STATUS REGISTER) 60 に記憶され、また受領 (RECEIVE) として送信元の配列演算ユニット 100 に返送される。受領は複数信号の送信元のフラグエンコーダ 59 で受信され、結果として複数信号の送信完了が確認される。アドレスによってステータスレジスタ 60 が選択されると、ステータスレジスタ 60 の内容がデータバス 52 を介してデータとしてプロセッサ 101 に送信される。1 つ以上の入力画像 (INPUT IMAGE) に対応した 1 つ以上の前入力送達 (FRONT INPUT SEND) をフラグエンコーダ 59 が受信すると 1 つ以上の入力画像からなる前入力データ群 (FRONT INPUT DATA SET) が必要な記憶容量分用意された前入力データレジスタ (FRONT INPUT DATA REGISTER) 61 に読み込まれる。アドレスによって前入力データレジスタ 61 が選択されると、前入力データレジスタ 61 の内容がデータとしてプロセッサ 101 に送信される。プロセッサ 101 が計算を完了したら、アドレスによって結果データレジスタ (RESULT DATA REGISTER) 62 が選択され、結果データレジスタ 62 が出力画像の画像データを結果データ (RESULT DATA) として読み込む。これと同時に、フラグエンコーダ 59 が結果送達 (RESULT SEND) を送信する。

【0254】近傍の配列演算ユニット 100 から計算に必要なデータを求められたら、アドレスとして出力データレジスタ (OUTPUT DATA REGISTER) 63 を選択し、近傍の配列演算ユニット 100 に送信すべきデータを計算データ (CALCULATION DATA) として出力データレジスタ 63 に読み込む。その後、隣接する全ての配列演算ユニット 100 に計算データとして送信される。上側の配列演算ユニット 100 から複数信号 (SIGNALS) を受信したら計算データを上入力データレジスタ (UPPER INPUT DATA REGISTER) 64 に読み込む。その後、アドレスにより上入力データレジスタ 64 が選択されたら、上入力データレジスタ 64 の内容が計算データとして送信される。下側、左側、右側の配列演算ユニット 100 から複数信号を受信した場合も同様であり、下入力データレジスタ 65、左入力データレジスタ 66、右入力データレジスタ 67 が同様動作する。

【0255】各種バッファ、各種レジスタ、アドレスデコーダ 54 の各ブロックは汎用的な電子回路である。フラグデコーダ 58 とフラグエンコーダ 59 は具体的には図 28 と図 29 に示すような入出力信号を有する。種別 (TYPE) は出力データレジスタ (OUTPUT DATA REGISTER) 63 に読み込まれた内容の種類を 5 ビットで表す。このビット数は配列演算ユニット 100 が受受信すべき全ての計算データを区別するのに十分な値である。カウント-X (COUNT-X) 及びカウント-Y (COUNT-Y) は各々 4 ビットの符号なし整数を表し、配列演算ユニット 100 の間の転送回数を示す。配列演算ユニット 100 が

計算データを送信する場合には各々のカウントが 0 となり、左右の配列演算ユニット 100 から送信された計算データを再度送信する場合にはフラグエンコーダ 59 のカウント-X に不足した値となり、上下の配列演算ユニット 100 から送信された計算データを再度送信する場合にはフラグエンコーダ 59 のカウント-Y に不足した値となる。プロセッサ 101 が上下左右のうちの方向に出力データレジスタ 63 の内容を送信するかをフラグレジスタ 57 の送達フラグ (SEND FLAG) に指定した後で、出力データレジスタ 63 を指定するアドレスデコーダ 54 の中央デコーディング (CENTRAL DECODING) をフラグデコーダ 58 が受信すると、フラグデコーダ 58 が送達 (SEND) を送達フラグの指定方向に合わせて出力する。送達フラグは 4 ビットで表し、配列演算ユニット 100 の計算データを四方の配列演算ユニット 100 に送信する場合にはプロセッサ 101 が 1111 と設定し、右側の配列演算ユニット 100 から送信された計算データを上下左側に転送する場合はプロセッサ 101 が 1110 と設定し、左側から上下右側に転送する場合は 1101 と設定し、下側から上側に転送する場合は 1000 と設定し、上側から下側に転送する場合は 0100 と設定する。これにより、転送に重複がなく効率的に転送できただけでなく、転送方向の決定規則が明確になっているので、種別、カウント-X 及びカウント-Y を組み合わせることにより、フラグエンコーダ 59 はどの配列演算ユニット 100 からどの種類の計算データが送信されたかを判定することができる。結果データレジスタ 62 に計算データが結果データとして読み込まれると同時にフラグデコーダ 58 は、結果デコーディング (RESULT DECODING) を受信し、結果送達 (RESULT SEND) を送信する。

【0256】フラグエンコーダ 59 は四方のうちいずれかでも送達を受信したら、受信方向の種別とカウント-X、カウント-Y を受信し、その部分のステータスレジスタ 60 の内容を更新する。この更新と同時に受信方向に受領を 1 にして送信する。送信元の配列演算ユニット 100 のフラグエンコーダ 59 では受領が 1 になった瞬間に受信し、ステータスレジスタ 60 の受領ステータス (RECEIVE STATUS) を更新する。これにより各配列演算ユニット 100 ではプロセッサ 101 がステータスレジスタ 60 の受領ステータスを検査するだけで、どの入力データレジスタに有効な計算データが記憶されているか判断することができる。そこで例えば上入力データレジスタ 64 に計算データが読み込まれていれば、プロセッサ 101 がアドレスを指定することにより上入力データレジスタ 64 からデータを読み込むことができるが、同時にアドレスデコーダ 54 から上デコーディング (UPPER DECODING) がフラグエンコーダ 59 に送信され、受領ステータスのうち上部分が 0 に戻され、上側に向けた受領が 0 として送信される。下左右側の場合も同様に動作

する。フラグエンコーダ59が1つでも入力画像用の前入力送達を受信したら、ステータスレジスタ60のうち受信した前入力送達に対応する入力画像用の前入力送達ステータス (FRONT INPUT SEND STATUS) を1にする。またプロセッサ101が入力画像用の前入力データレジスタ61からデータを読み込むとき、アドレスデコーダ54がフラグエンコーダ59に前デコーディング (FRONT DECODING) を送信し、受信した前入力送達に対応する前入力送達ステータスを0にする。プロセッサ101はステータスレジスタ60の内容を読み込むことにより、前入力データレジスタ61に最新の入力画像が記憶されているかどうか判断することができる。

【0257】プロセッサ101がコントローラ103を介して四方の配列演算ユニット100に計算データを送信する場合のアルゴリズムを図30に示す。図30は、プロセッサ101によるプログラム制御と、フラグデコーダ58及びフラグエンコーダ59によるハードウェアロジックとの混成による処理を示すものである。図30に対して、ステップ71では、プロセッサ101がステータスレジスタ60の内容を読み込む。ステップ72では、読み込んだ内容のうち受領ステータスが全て0であるかを判断する。NOなら処理を終了する。YESならステップ73に移行する。ステップ73では、プロセッサ101が隣接する配列演算ユニット100に送信するデータの種別とカウンタと送信方向を決定し、その内容をフラグレジスタ57に書き込む。ステップ74では、プロセッサ101が隣接する配列演算ユニット100に送信するデータを出力データレジスタ63に書き込む。ステップ75では、出力データレジスタ63の内容を計算データとして、隣接する配列演算ユニット100に送信する。ステップ76では、フラグレジスタ57の送達フラグで指定された方向にのみ送達を1にして送信する。これによりプロセッサ101の1回の送信アルゴリズムは終了する。プロセッサ101は、送信すべきデータがメモリ102内で更新される度にこの送信アルゴリズムを開始する。

【0258】コントローラ103が上側の配列演算ユニット100から計算データを受信する場合のアルゴリズムを図31に示す。図31は、フラグデコーダ58及びフラグエンコーダ59によるハードウェアロジックによる処理を示すものである。図31に対して、ステップ81では、フラグエンコーダ59が送達を入力する。ステップ82では、送達が1であるかをフラグエンコーダ59が判断する。NOなら処理を終了する。YESならステップ83に移行する。ステップ83では、上入力データレジスタ64が上側から送信された計算データを読み込む。ステップ84では、フラグエンコーダ59がステータスレジスタ60のうち上側用の受領ステータスを1にすると同時に受領を1にして上側の配列演算ユニット100に送信する。下左右側の場合も同様である。

これによりコントローラ103の1回の受信アルゴリズムは終了する。コントローラ103は常時上下左右の配列演算ユニット100からの送達を監視し、この送達を受信する度にこの受信アルゴリズムを開始する。

【0259】プロセッサ101が上入力データレジスタ64からデータを受信する場合のアルゴリズムを図32に示す。図32は、プロセッサ101によるプログラム制御と、フラグデコーダ58及びフラグエンコーダ59によるハードウェアロジックとの混成による処理を示すものである。図32に対して、ステップ91では、プロセッサ101がステータスレジスタ60の内容を読み込む。ステップ92では、読み込んだ内容のうち上側用の受領ステータスが1であるかを判断する。NOなら処理を終了する。YESならステップ93に移行する。ステップ93では、プロセッサ101が上入力データレジスタ64からデータを読み込む。ステップ94では、フラグエンコーダ59がステータスレジスタ60のうち上側用の受領ステータスを0にすると同時に受領を0にして上側の配列演算ユニット100に送信する。下左右側の場合も同様である。これによりプロセッサ101の1回の受信アルゴリズムは終了する。プロセッサ101は一定間隔でステータスレジスタ60の内容を監視し、上下左右いずれかの受領ステータスが1である度にこの受信アルゴリズムを開始する。なおここではプロセッサ101が一定間隔でステータスレジスタ60の内容を常時監視する場合を示したが、コントローラ103に割り込み回路を付加することにより、プロセッサ101は割り込みによりこの受信アルゴリズムを開始することもできる。

【0260】なおこの配列演算ユニット100は、主に1つ以上の入力画像から1つの出力画像を生成することを前提に記述したが、用途に応じて計算途中の計算データを出力できるように回路を変更する必要がある。その際には、出力すべき計算データの数だけフラグデコーダ58の結果送達を増やし、結果データレジスタ62に読み込まれた計算データに対応する結果送達のみを1にするようにプログラムを変更するだけで良い。

【0261】以上、本実施形態を説明したが、本発明は上述の実施形態には限定されることがなく、当業者であれば種々なる態様を実施可能であり、本発明の技術的思想を逸脱しない範囲において本発明の構成を適宜変更できることは当然であり、このような変更も、本発明の技術的範囲に属するものである。

#### 【0262】

【発明の効果】請求項1及び2記載の発明によれば、大規模集積回路 (LSI) 及び圧力センサ2の製造メーカーが、この製造メーカーが保有するLSI及び圧力センサ製造技術をそのまま用いて高機能タッチセンサを製造することができる。したがって製造メーカーは短期間しかも安価に高機能タッチセンサを製造できるので、本



発明の利用者は、家屋、コンビニエンスストア、スーパー、事務所、工場、一般道路、駐車場、駅、公益機関、車両、船舶、及び宇宙ステーションにおいて、入力装置、保持装置、検査装置、保安装置、警戒装置、計数装置、ベットロボット、受付ロボット、作業ロボット、及び福祉ロボットなど色々な用途に高機能タッチセンサを利用できるようになる。特に圧力センサ2が圧力の微妙な変化を捉えることができれば、本発明は筆圧、指圧及び指紋の凹凸を分類することができ、利用者は本発明を個人識別にも利用することができる。

【0263】請求項3記載の発明によれば、LSI及び圧力センサ2の設計者は、1個の機能コラム1を設計して二次元格子状に配列するだけで、適当なサイズの画像を射影することができ、高機能タッチセンサを容易に設計することができる。したがってLSI及び圧力センサ2の設計者は、短期間でしかも安価に高機能タッチセンサを設計することができる。

【0264】請求項4記載の発明によれば、大規模集積回路(LSI)及び温度センサの製造メーカーは、この製造メーカーが保有するLSI及び温度センサ製造技術をそのまま用いて高機能タッチセンサを製造することができる。したがって製造メーカーは短期間にしかも安価に高機能タッチセンサを製造できるので、本発明の利用者は、家屋、コンビニエンスストア、スーパー、事務所、工場、一般道路、駐車場、駅、公益機関、車両、船舶、及び宇宙ステーションにおいて、入力装置、保持装置、検査装置、保安装置、警戒装置、計数装置、ベットロボット、受付ロボット、作業ロボット、及び福祉ロボットなど色々な用途に高機能タッチセンサを利用できるようになる。特に温度センサが温度の微妙な変化を捉えることができれば、本発明は体温及び指の汗の分布を分類することができ、利用者は本発明を個人識別にも利用することができる。

【0265】請求項5記載の発明によれば、大規模集積回路(LSI)及び電極の製造メーカーは、この製造メーカーが保有するLSI及び電極製造技術をそのまま用いて高機能タッチセンサを製造することができる。したがって製造メーカーは短期間にしかも安価に高機能タッチセンサを製造できるので、本発明の利用者は、家屋、コンビニエンスストア、スーパー、事務所、工場、一般道路、駐車場、駅、公益機関、車両、船舶、及び宇宙ステーションにおいて、入力装置、保持装置、検査装置、保安装置、警戒装置、計数装置、ベットロボット、受付ロボット、作業ロボット、及び福祉ロボットなど色々な用途に高機能タッチセンサを利用できるようになる。特に電極が水分の微妙な変化を捉えることができれば、本発明は指の表面抵抗、誘電率及び汗の分布を分類することができ、利用者は本発明を個人識別にも利用することができる。

【図面の簡単な説明】

【図1】1個の圧力センサ、1個のA/D変換回路、及び1個の配列演算ユニットを備えた機能コラムのブロック図である。

【図2】1個の圧力センサ、1個のA/D変換回路、1個の配列演算ユニット、及び1個の発振回路を備えた機能コラムのブロック図である。

【図3】圧力の位置及び大きさを検知する装置用の機能コラムのブロック図である。

【図4】発振回路を搭載し、圧力の変化を抽出して圧力の位置及び大きさを検知する装置用の機能コラムのブロック図である。

【図5】二次元格子状に機能コラムを配列した場合のタッチセンサのブロック図である。

【図6】機能コラムを大規模集積回路の実装面に対して垂直に実装した場合のタッチセンサのブロック図である。

【図7】二次元格子状に配列された配列演算ユニットのブロック図である。

【図8】本実施形態の画像記憶手段のアルゴリズムを示すフローチャートである。

【図9】本実施形態の画像振動手段のアルゴリズムを示すフローチャートである。

【図10】本実施形態のエッジ情報生成手段のアルゴリズムを示すフローチャートである。

【図11】デジタル画像を用いて粗エッジ情報を形成エッジ情報に形成する場合の説明図である。

【図12】本実施形態のエッジ情報形成手段のアルゴリズムを示すフローチャートである。

【図13】デジタル画像の切出領域を正規化する場合の説明図である。

【図14】本実施形態の領域正規化手段のアルゴリズムを示すフローチャートである。

【図15】エッジ情報画像中の物体の位置及び大きさを検出する場合の説明図である。

【図16】本実施形態の位置/大きさ検出手段のアルゴリズムを示すフローチャートである。

【図17】物体領域画像中の物体の位置及び大きさを検出する場合の説明図である。

【図18】本実施形態のデジタル画像用の画像保持手段のアルゴリズムを示すフローチャートである。

【図19】正規化画像に対してテンプレート画像の中からパターンマッチングをする場合の説明図である。

【図20】本実施形態のパターンマッチング手段のアルゴリズムを示すフローチャートである。

【図21】三角形のエッジ情報が三角形の内側領域と外側領域に分離する状態を示す説明図である。

【図22】本実施形態の物体/背景分離手段のアルゴリズムを示すフローチャートである。

【図23】破線状態の三角形のエッジ情報が破線三角形の内側領域と外側領域に分離する状態を示す説明図であ

る。

【図24】三角形を2つ重ねたエッジ情報が2つの三角形領域と背景領域に分離する状態を示す説明図である。

【図25】2つの円形物体領域を重ねた時の破線状態のエッジ情報が2つの円形領域と背景領域に分離した状態を示す説明図である。

【図26】配列演算ユニットの内部構造のブロック図である。

【図27】コントローラのブロック図である。

【図28】フラグデコーダの入出力信号を示す説明図である。

【図29】フラグエンコーダの入出力信号を示す説明図である。

【図30】プロセッサがコントローラを介して隣接する配列演算ユニットにデータを送信するアルゴリズムを示すフローチャートである。

【図31】コントローラが隣接する配列演算ユニットからデータを受信するアルゴリズムを示すフローチャートである。

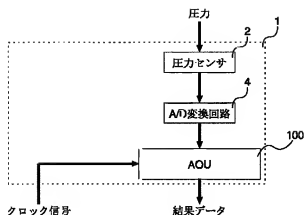
【図32】プロセッサが上入力レジスタからデータを受信するアルゴリズムを示すフローチャートである。

【符号の説明】

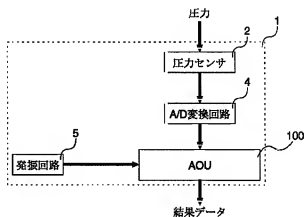
- 1 機能コラム
- 2 圧力センサ
- 4 A/D変換回路
- 5 発振回路
- 12 画像記憶手段
- 14 エッジ情報生成手段
- 15 エッジ情報形成手段
- 16 物体/背景分離手段
- 17 位置/大きさ検出手段
- 27 領域正規化手段
- 29 パターンマッチング手段
- 38 画像保持手段
- 51 アドレスバス
- 52 データバス
- 53 アドレスバッファ
- 54 アドレスデコーダ
- 55 データバッファ
- 56 内部データバス
- 57 フラグレジスタ
- 58 フラグデコーダ
- 59 フラグエンコーダ

- 60 ステータスレジスタ
- 61 前入力データレジスタ
- 62 結果データレジスタ
- 63 出力データレジスタ
- 64 上入力データレジスタ
- 65 下入力データレジスタ
- 66 左入力データレジスタ
- 67 右入力データレジスタ
- 100 配列演算ユニット
- 101 プロセッサ
- 102 メモリ
- 103 コントローラ
- 110 データ処理装置
- 111 デジタル画像
- 112 粗エッジ情報
- 113 粗エッジ情報画像
- 114 形成エッジ情報
- 115 形成エッジ情報画像
- 131 重複情報
- 132 重複情報画像
- 141 物体領域
- 142 物体領域画像
- 143 分離物体領域
- 144 正規化領域
- 145 正規化画像
- 146 テンプレート画像
- 147 マッチング結果画像
- 151 三角形のエッジ情報
- 152 三角形の内側領域
- 153 三角形の外側領域
- 154 破線状態の三角形のエッジ情報
- 155 破線三角形の内側領域
- 156 破線三角形の外側領域
- 157 前方の三角形のエッジ情報
- 158 後方の三角形のエッジ情報
- 159 前方三角形の内側領域
- 160 後方三角形の内側領域
- 161 二重三角形の背景領域
- 162 2つの重なった円形のエッジ情報
- 163 前方円形の内側領域
- 164 後方円形の内側領域
- 165 二重円の背景領域

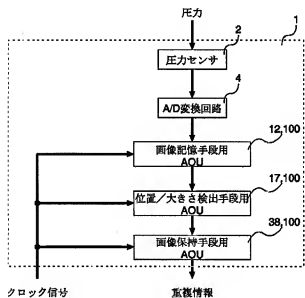
【図1】



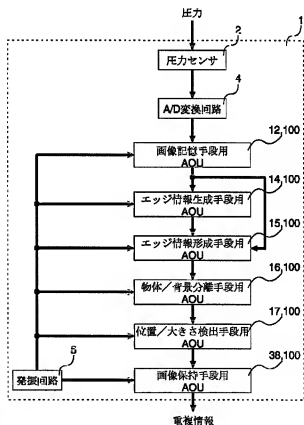
【図2】



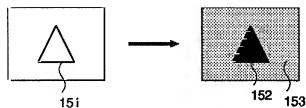
【図3】



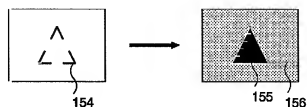
【図4】



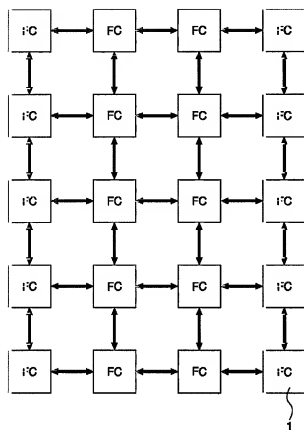
【図21】



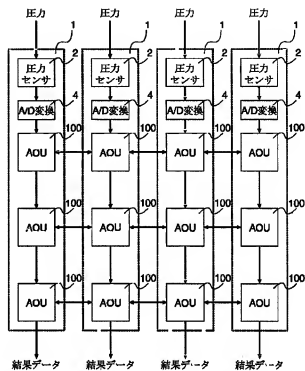
【図23】



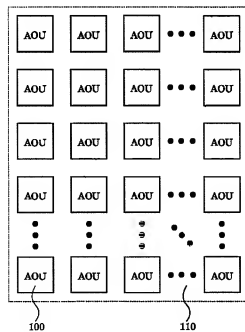
【図5】



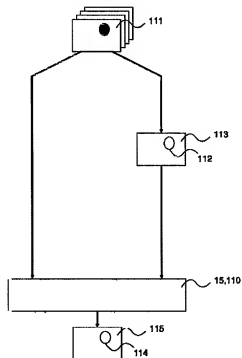
【図6】



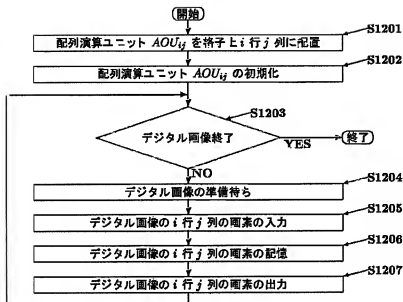
【図7】



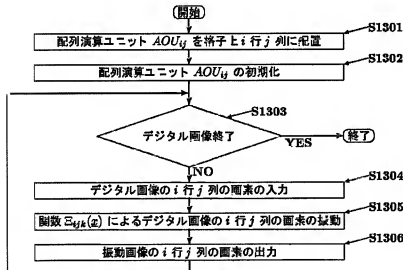
【図11】



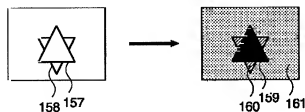
【図8】



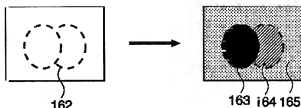
【図9】



【図24】



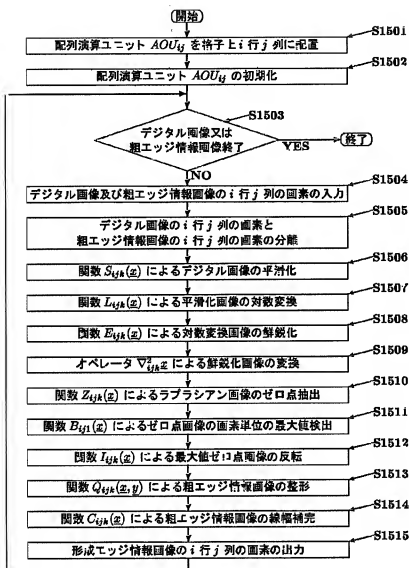
【図25】



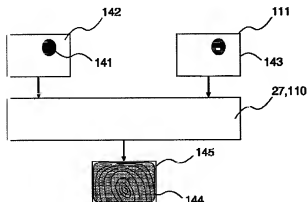
【図10】



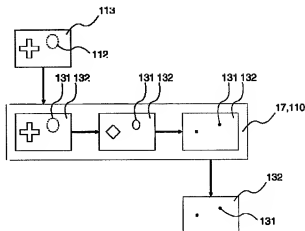
【図12】



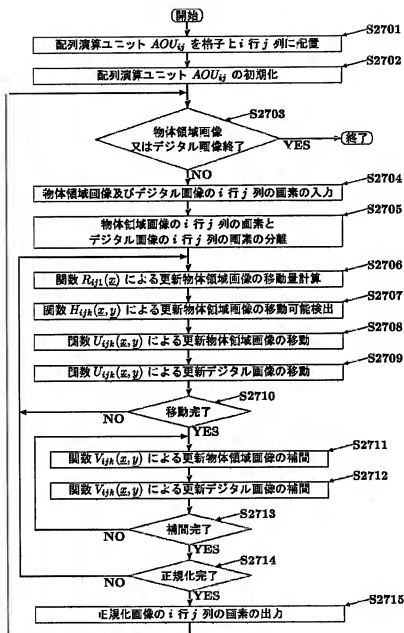
【図13】



【図15】

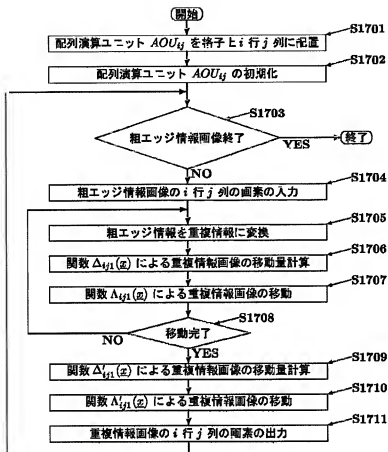


【図14】

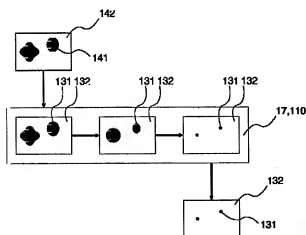




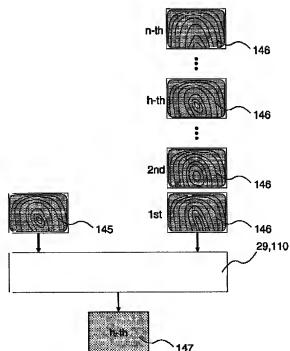
【図16】



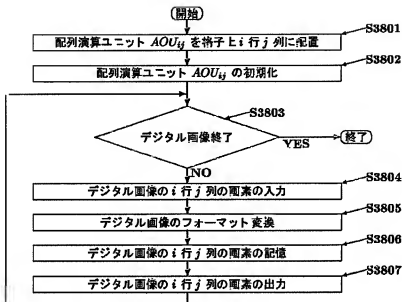
【図17】



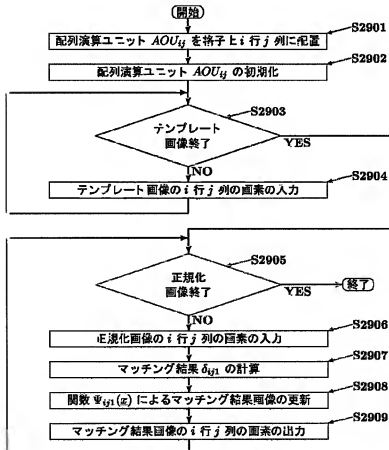
【図19】



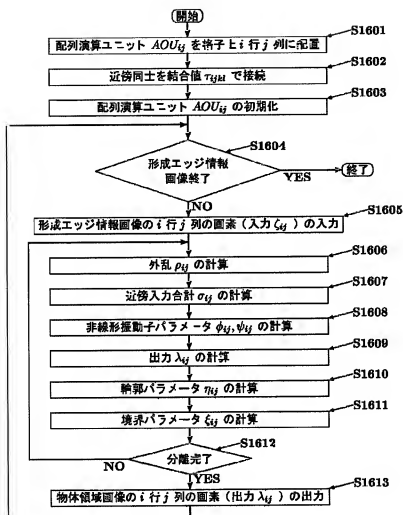
【図18】



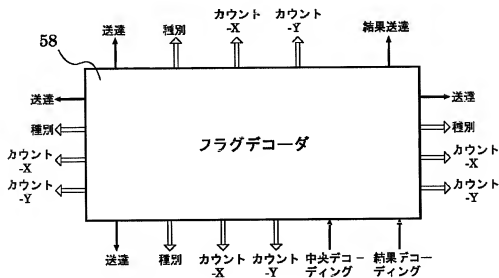
【図20】



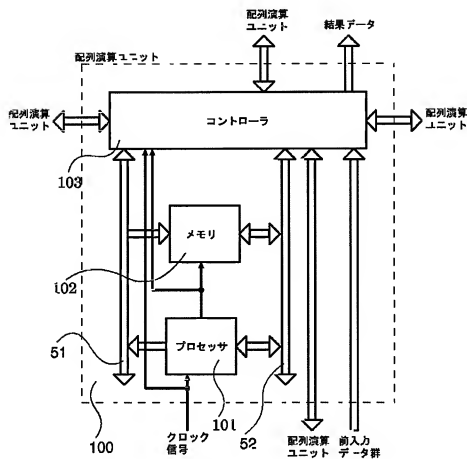
【図22】



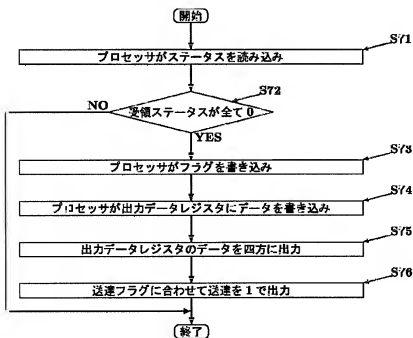
【図28】



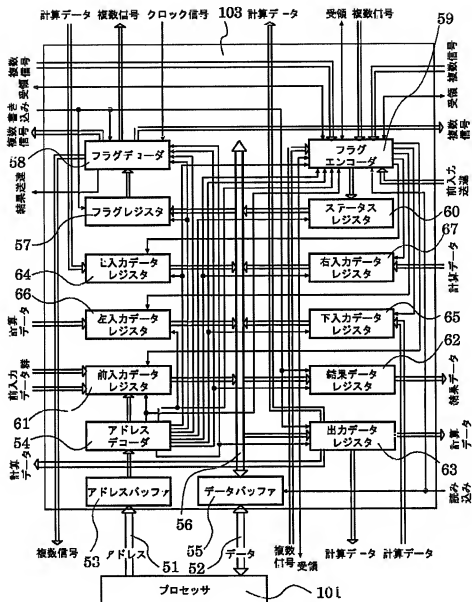
【図26】



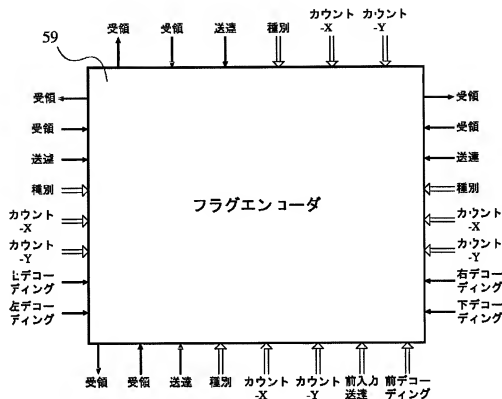
【図30】



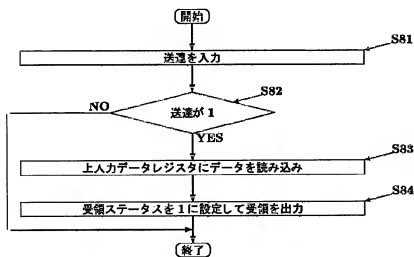
【図27】



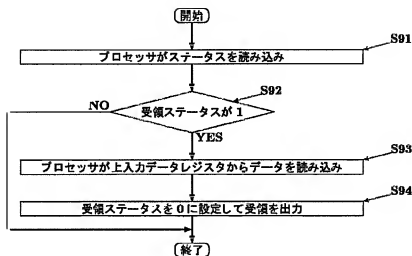
【図29】



【図31】



【図32】




---

フロントページの続き

Fターム(参考) 2F051 AA21 AB03 AB04 AB06 AC01

AC09 BA07

2F063 AA03 AA41 AA48 BA29 BA30

BB01 BD01 BD05 CB00 DA02

DD07 EB02 EC00 KA01 LA12

LA18 LA19 LA23 LA26 LA30

NA02 NA06

2F069 AA03 AA13 AA64 AA96 GG02

GG06 GG16 GG20 GG58 GG59

HH30 NN00 NN08 NN25